Group Key Establishment in Wireless Ad Hoc Networks *

Eric Ricardo Anton　　　Otto Carlos Muniz Bandeira Duarte

`eric@gta.ufrj.br`　　　`otto@gta.ufrj.br`

Grupo de Teleinformática e Automação

COPPE/EE – Programa de Engenharia Elétrica

Universidade Federal do Rio de Janeiro – Brazil

`http://www.gta.ufrj.br/`

Abstract

This paper analyzes security issues of wireless ad hoc networks. This new environment, unlike traditional networks, has no infrastructure and the nodes depend on each other to keep the network connected. Key establishment protocols are presented and analyzed. A proposal for group key establishment among the members of an ad hoc network is presented and is the base for the deployment of security services.

Keywords: Computer Networks, Network Security, Key Management, Wireless Ad Hoc Networks.

1 Introduction

Applications in wireless networks have reached an extraordinary success in the last years. This success is due to its intrinsic feature of not possessing wires, that facilitates the installation, extension and maintenance of a network and, mainly, allows mobility. A whole new generation of portable devices is commercially available, such as cellular telephones, digital personal assistants (PDA), auricular phones, etc. On the other hand, wireless communications are usually transmitted by radio frequencies, which are highly vulnerable, since it is possible for an unauthorized party, located within the transmitter's communication radius, to listen to the communication. Thus, wireless communications need security mechanisms in order to guarantee the integrity and the privacy of the communication, as well as the authentication of the entities involved.

Wireless networks can be classified as infra-structured or infra-structureless. Infra-structured networks are characterized by mobile devices communicating through one or more centralizing devices, called access points. An example of this kind of network is the cellular telephone network, in which each telephone communicates with a radio-base station, but never directly with another telephone.

In infra-structureless networks, also called ad hoc networks, all devices are able to establishing direct communication with other devices that are within its communication range. There is no centralizing entity like the access point. Ad hoc networks apply to scenarios where some devices are confined in an environment unprovided of any network infrastructure. Examples of infra-structureless environments are islands, places of difficult access (mountains, forests,

---

deserts, glaciers, etc), events (artistic shows, assemblies, etc) or emergency situations (floodings, earthquakes, lack of energy, rescue of hostages in assaults and kidnappings, etc). An ad hoc network can be established and an electronic meeting can easily take place. In this case it can be assumed that all devices are static or have little mobility, and that they are mutually reachable. Thus, there is no accessibility problem and the devices can communicate directly.

Multihop communications are another possibility. Two devices that are mutually unreachable can communicate as long as there is at least one chain of devices that is reachable by both. In this kind of network, in contrast to traditional networks, all devices are capable of routing packages. A possible scenario where multihop ad hoc networks can be useful consists of several devices, static or with little mobility, where the devices' communication range is extended by using other devices as simple repeaters. A possible application is the establishment of a wireless network connecting kiosks that are placed in such a way that the distance between two of them is greater then the transmission radius of the wireless devices.

A complex scenario would be a wireless network connecting mobile devices spread over an area greater than the devices' transmission radius. This implies multihop ad hoc networks in which the accessibility of all devices is not guaranteed. Due to mobility, the set of reachable stations is dynamic, as it varies through time. This implies that the devices must be capable of routing messages and that the routing tables are constantly updated, since they reflect the devices' instantaneous locations. An application of this scenario is a war operation where soldiers move and a network is dynamically installed and reconfigured. Another possible application is a police operation to arrest criminals.

This paper approaches the problem of establishing safe group communication among wireless devices on an ad hoc network. The problem of establishing a shared secret key among the members of a group will be analyzed. Once the group members know the secret key, they are able to encrypt and decrypt messages to each other and transmit these messages securely. Only the secret group key establishment and management are analyzed. The multicast communication, the ad hoc routing and the congestion and error control are not within the scope of this work.

The remainder of this paper is organized as follows. Section 2 presents the main concepts of group key establishment and the main group key establishment protocols. In Section 3 group key establishment in ad hoc networks is discussed and a protocol for group member discovery is proposed. Section 4 presents final comments on this work.

2   Group key management

Group key management includes activities for establishment and maintenance of the group key. Maintenance activities consist of changing the group key due to group members addition or exclusion or due to the use of the group key for long periods of time (key refresh). A good key management policy is extremely important for the deployment of security services.

The group key establishment can be centralized, also called distributive, where an entity is responsible for generating the group key and distributing it to the other group members. This approach has the advantage of being simple. In a distributed, or contributory, key establishment all group members contribute for the group key generation. This approach is fault tolerant and diminishes the risks of vicious key generation by a single entity. A variation of this last approach consists of a partially contributory approach that allows for a subgroup to be responsible for generating the group key. In [1] an architecture is proposed in which a key management

service is distributed among various nodes and the consensus of a minimum number of nodes is necessary.

A group can be static or dynamic. A dynamic group allows the exclusion of members as well as the addition of new members. Key management for dynamic groups can provide forward secrecy, when members that leave the group are unable to compute future group keys, and backward secrecy, when new group members are unable to compute old group keys.

In this paper some distributed group key establishment protocols that provide backward and future secrecy are presented. The Diffie-Hellman algorithm, which is the main algorithm for secret key establishment between two entities, and some of the main group key establishment protocols for traditional networks are presented. These protocols are natural extensions of the Diffie-Hellman algorithm to the multiparty case, as defined in [2], inherit all its security characteristics and provide a contributory group key establishment.

2.1 The Diffie-Hellman Algorithm

Developed by Diffie and Hellman [3], this algorithm allows the establishment of a cryptographic secret key between two entities by means of data exchange through an insecure communication channel. The algorithm executed between two entities A and B is defined as follows:

- A and B agree[1] on two randomly chosen numbers p e g, so that p is a large prime number and $g < p$;

- A chooses a secret random number S_A and B chooses a secret random number S_B;

- A computes a public value $T_A = g^{S_A} \bmod p$ and B computes a public value $T_B = g^{S_B} \bmod p$;

- A sends T_A to B and B sends T_B to A;

- A computes $T_B{}^{S_A} \bmod p = (g^{S_B})^{S_A} \bmod p$ and B computes $T_A{}^{S_B} \bmod p = (g^{S_A})^{S_B} \bmod p$.

Since $(g^{S_A})^{S_B} \bmod p = (g^{S_B})^{S_A} \bmod p = K$, these two entities share a secret cryptographic key K.

In other works, two entities are able to exchange information through a channel that anyone can listen to and at the end of the process the two entities, and only the two entities, share the same secret key.

The security of this algorithm is based on the difficulty to calculate the secret key $K = g^{S_A S_B} \bmod p$, despite of knowing the public values $g^{S_A} \bmod p$ and $g^{S_B} \bmod p$, when the prime number p is sufficiently large.

This algorithm has a weakness that consists on the lack of authentication between the two entities. Even though they are able to establish a secret key, there is no guarantee that these entities are who they claim to be.

---

[1]All information exchanged between these entities use an insecure channel, being therefore of public knowledge.

2.2 The Ingemarsson et al. Protocol

This protocol, presented by Ingemarsson *et al.* [4], was one of the first attempts to extend the Diffie-Hellman algorithm to group communications. The n group members must be arranged in a logical ring. At the beginning of the protocol execution, each participant M_i, $i \in [1, n]$ generates a random value S_i that is its contribution for the group key. At each round, each participant raises the value received in the previous round to its secret value S_i and sends it to the next node in the sequence. Therefore n messages are exchanged among the n group members at each round. After $n - 1$ rounds every node computes the same group key $K_n = g^{S_1 \cdot S_2 \cdot \ldots \cdot S_n}$. The main disadvantages of this approach are the high number of messages exchanged and the high number of exponential operations executed (see Table 1 and Figure 2).

2.3 The Burmester and Desmedt Protocol

This protocol was presented by Burmester and Desmedt [5] and is executed in three rounds. Each participant M_i, $i \in [1, n]$ executes the following operations:
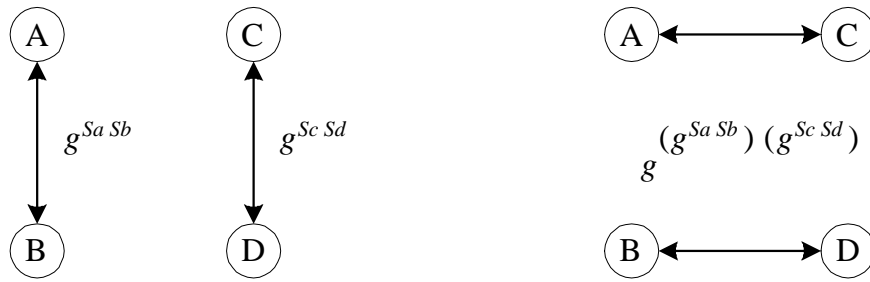
- generates a secret random value S_i and broadcasts $z_i = g^{S_i}$ to the other participants;

- computes and broadcasts $X_i = (\frac{z_{i+1}}{z_{i-1}})^{S_i}$ to the other participants;

- computes the group key $K_n = z_{i-1}^{nS_i} \cdot X_i^{n-1} \cdot X_{i+1}^{n-2} \cdot \ldots \cdot X_{i-2} \bmod p$.

This group key has the form $K_n = g^{S_1 \cdot S_2 + S_2 \cdot S_3 + \ldots + S_n \cdot S_1}$ and shares the security characteristics presented by the Diffie-Hellman algorithm. This protocol is efficient with respect to the total number of rounds. This characteristic could allow faster execution, but each round requires n simultaneous broadcasts. Simultaneous broadcasts are usually not possible, even in wireless networks, because there can be only one broadcast message at a given moment. Due to this characteristic, the deployment of this protocol must use sequential broadcast messages. Since each broadcast message acts like a round, there is no longer a low number of rounds advantage. Another disadvantage is that this protocol makes use of a high number of exponential operations (see Table 1 and Figure 2).

2.4 The Hypercube and Octopus Protocols

The Hypercube protocol was presented by Becker and Willie [6] and intents to overcome the high number of messages needed by the protocol proposed by Ingemarsson *et al.* by logically arranging the nodes in a hypercube. For a network consisting of four nodes positioned as a square, a key is established between A and B ($g^{S_a S_b}$), and another key between C and D ($g^{S_c S_d}$). These keys are used to establish a single key ($g^{(g^{S_a S_b})(g^{S_c S_d})}$) among the four entities, as presented in Figure 1. This behavior can be generalized for higher numbers of nodes, as long as the number of participants equals 2^d, for $d \in \mathbb{I}$. This protocol executes in d rounds.

The Octopus protocol is an extension of the Hypercube protocol for networks with an arbitrary number of nodes. A subgroup of nodes is arranged in a hypercube, composing a core. Each core node establishes a key with each nearby non-core node using the Diffie-Hellman protocol. The product of these keys is used to establish a key among the core nodes as specified by the Hypercube protocol. This key is then distributed to the other nodes.

Figure 1: Hypercube protocol for $n = 4$.

2.5 The CLIQUES protocol suite

Developed by Steiner *et al.* [2, 7, 8, 9, 10], the CLIQUES protocol suite consists of key management protocols for dynamic groups. Two of these protocols, IKA.1 e IKA.2 (Initial Key Agreement 1 and 2), are defined for group key establishment. Other protocols are specified for member and subgroup addition and exclusion and key refresh.

The protocols from this suite can provide member authentication, which solves the Diffie-Hellman authentication vulnerability mentioned in Section 2.1.

2.5.1 IKA.1

The IKA.1[2] protocol executes in two stages:

1. $M_i \Rightarrow M_{i+1}$:

   - $\{g^{\frac{S_1 \cdot S_2 \cdot \ldots \cdot S_i}{S_k}} \mid k \in [1, i]\}$, $g^{S_1 \cdot S_2 \cdot \ldots \cdot S_i}$

2. $M_n \Rightarrow M_i$:

   - $\{g^{\frac{S_1 \cdot S_2 \cdot \ldots \cdot S_n}{S_i}} \mid i \in [1, n-1]\}$,

for $i \in [1, n-1]$.

At the first stage contributions are collected from all group members throughout $n - 1$ rounds. Each group member (except the first) receives a data set that represents the partial contributions from all the group members that have already executed this first stage. The member adds its contribution and sends a new data set to the next group member.

As an example, node M_4 receives the set $\{g^{S_1 \cdot S_2 \cdot S_3}, g^{S_1 \cdot S_2}, g^{S_1 \cdot S_3}, g^{S_2 \cdot S_3}\}$, and sends the set $\{g^{S_1 \cdot S_2 \cdot S_3 \cdot S_4}, g^{S_1 \cdot S_2 \cdot S_3}, g^{S_1 \cdot S_2 \cdot S_4}, g^{S_1 \cdot S_3 \cdot S_4}, g^{S_2 \cdot S_3 \cdot S_4}\}$ to M_5. The set sent by the ith node consists of i intermediate values, each containing $i - 1$ exponents, and a cardinal value contenting i exponents that correspond to the exponentiation base raised to every contribution generated so far.

The last group member, M_n, is called the group controller. At the end of the first stage it receives a data set whose cardinal value is $g^{S_1 \cdot S_2 \cdot \ldots \cdot S_{n-1}}$ and computes the group key $K_n = g^{S_1 \cdot S_2 \cdot \ldots \cdot S_n}$.

---

[2]This protocol was formerly known as GDH.2.

At the second stage, the group controller adds its contribution to each intermediate value and broadcasts this new data set to every other node in the network. Each intermediate value now consists of the contribution of all group members except one. In order to compute the group key, each group member M_i identifies the appropriate intermediate value (the one that does not contain its contribution) and raises it to its contribution S_i, obtaining K_n.

2.5.2 IKA.2

The IKA.1 protocol requires $i + 1$ exponential operations when executed by the ith node. In some environments it is desirable to minimize the computational effort demanded from each group member. Some examples of these environments are groups with a high number of members and groups whose members have limited computational capacity. The IKA.2[3] protocol was proposed in order to minimize the demanded computational cost. It is similar to the IKA.1 protocol but is executed in four stages:

1. $M_i \Rightarrow M_{i+1}$, $i \in [1, n-2]$:

   - $g^{S_1 \cdot S_2 \cdot \ldots \cdot S_i}$

2. $M_{n-1} \Rightarrow M_i$, $i \in [1, n-2]$:

   - $g^{S_1 \cdot S_2 \cdot \ldots \cdot S_{n-1}}$

3. $M_i \Rightarrow M_n$, $i \in [1, n-1]$:

   - $g^{\frac{S_1 \cdot S_2 \cdot \ldots \cdot S_{n-1}}{S_i}}$

4. $M_n \Rightarrow M_i$, $i \in [1, n-1]$:

   - $\{g^{\frac{S_1 \cdot S_2 \cdot \ldots \cdot S_n}{S_i}} \mid i \in [1, n-1]\}$.

At the first stage contributions are collected from the $n-2$ first group members by means of a single message sent from one member to the next that gathers all the previous contributions. At the second stage, M_{n-1} adds its contribution to the received message and broadcasts this new message to the $n-2$ first members. At the third stage each member factors out its own contribution and sends this result to the last group member. At the last stage, M_n collects all the sets from the previous stage, raises each one of them to its contribution S_n and broadcasts these results to the other group members, allowing them to compute the group key.

These two protocols have the advantage of requiring a low number of messages (see Table 1 and Figure 2). The IKA.2 protocol has reduced the number of exponential operations required for group key establishment. Unlike the other presented protocols, this protocol suite provides mechanisms for group addition and exclusion, making it unnecessary to execute the entire key establishment protocol. This characteristic reduces the involved costs and provides backward and forward secrecy.

---

[3]This protocol was formerly known as GDH.3.

2.6 Protocols comparison

Table 1 compares the presented group key establishment protocols[4]. This table presents for each protocol the total number of messages and the total number of exponential operations necessary and the need for synchronism among the participants at the moment a message is sent. Figure 2 graphically shows the information from Table 1.

	Messages	Exponential op.	Sync.
Ing.	$n(n-1)$	n^2	Yes
B-D	$2n$	$n(n-1)$	Yes
Hypercube	$n \cdot \log_2 n$	$2n \cdot \log_2 n$	Yes
IKA.1	n	$\frac{n(n+3)}{2} - 1$	No
IKA.2	$2n-1$	$5n - 6$	No

Table 1: Comparison of the presented group key establishment protocols.
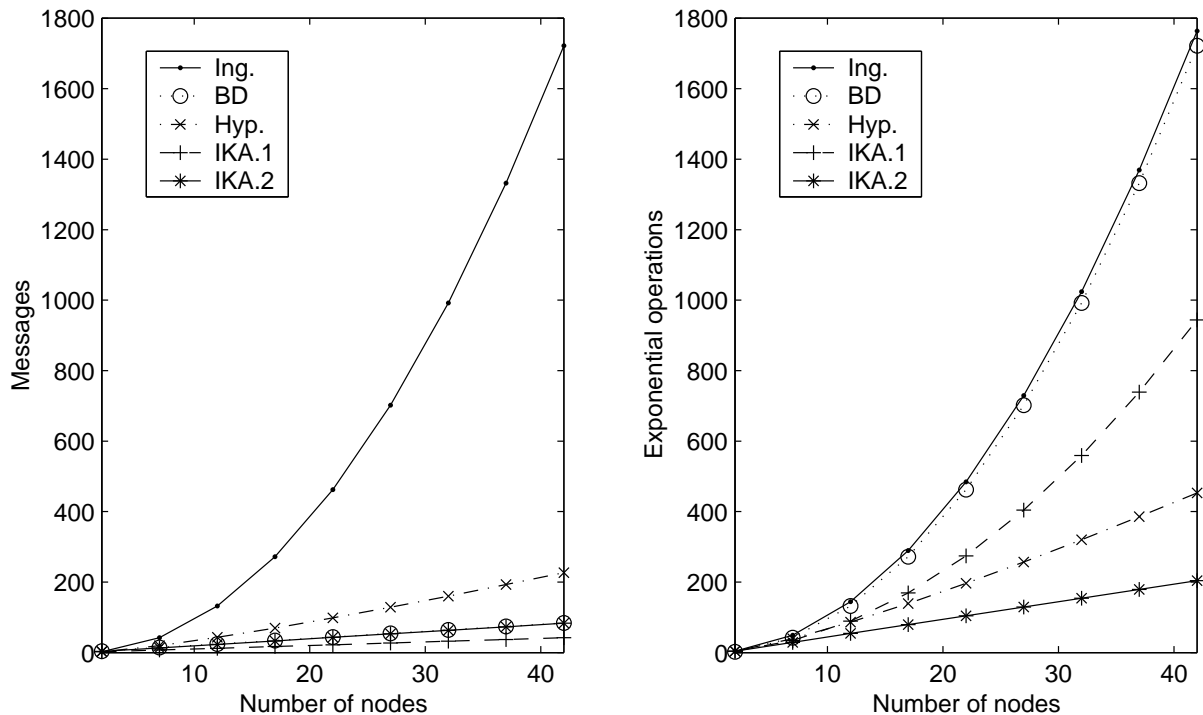

Figure 2: Number of messages and exponential operations as a function of the number of nodes for each protocol.

The IKA.1 protocol has the best number of messages performance, followed by the IKA.2 and the Burmester and Desmedt protocols. When the number of exponential operations is analyzed, the best performance is presented by the IKA.2 protocol, followed by the Hypercube and the IKA.1 protocols. The joint analysis of these two metrics points the IKA.1, IKA.2 and Hypercube protocols as the most interesting. The Ingemarsson *et al.* protocol presents bad

---

[4]The Octopus protocol was excluded from this analysis because the number of messages and exponential operations depend on the number of nodes in the core.

performance for both metrics. The Burmester and Desmedt protocol has a bad number of exponential operations performance. Its number of messages performance is good but equivalent to that of the IKA.2 protocol.

3   Group key establishment in wireless ad hoc networks

The use of centralized group key management is not convenient in an ad hoc environment due to factors like possible server unavailability. This unavailability might be temporary, when the node moves away from the others, or permanent, when its battery supply runs out and there is no way to recharge it. A central server might still become a bottleneck node in the network, minimizing the network performance.

From the protocols presented in Section 2, the protocols from the CLIQUES protocol suite are among the ones with best performance, do not need synchronism among the group members, and are the only ones to provide mechanisms for group members addition and exclusion and group member authentication. These reasons led to the choice of the CLIQUES protocol suite as the most appropriate for secret key management in wireless ad hoc networks. This protocol suite requires a sequencing among the group members that defines in particular the sequence of group members the contributions must go through and the last node in the sequence, that acts as the group controller. The way by which this sequencing must be established is not defined and can be left to the implementor. This sequencing can be fixed, using a predefined sequence, or be determined during the execution of the group key establishment protocol.

In an ad hoc network with constant node mobility the use of a single predefined sequence can be inefficient if this sequence does not correspond to the best geographic node placement. As an example, a fixed sequence can lead to communications through extensive node chains, generating inefficient message forwardings, as illustrated by Figure 3. The group key establishment would be more efficient if the node sequence 1-5-3-2-4-6 was used. This approach demands an *a priori* knowledge of all nodes that will set up the group which is inappropriate in some scenarios. There is also the possibility of protocol deadlock if a group member is unavailable for a long period of time during its turn to send or receive the group key contribution. This situation prevents the protocol from completing until the connectivity with neighbor nodes is reestablished.
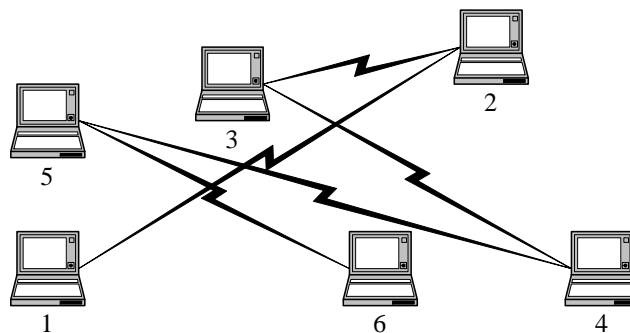

Figure 3: Unnecessary message exchange in CLIQUES.

3.1 A protocol for group member discovery

This paper proposes a protocol for establishing a sequence among the group members during the execution of the IKA.1 or IKA.2 protocols. When a node wishes to send its contribution to the next node in the sequence, it searches for other group members and chooses one to be the next node in the sequence. The proposed search is based on ad hoc routing protocols and makes use of flooding[5] messages to find other group members. Every node that receives this message and has not yet participated in the group key establishment must send a reply message. If the searching node does not receive any reply within a specified period of time, it can try again or assume there are no unknown members left in the network and finalize the group key establishment by becoming the group controller M_n. If reply messages are received, one of the replying nodes is chosen as the next node in the key contribution sequence.

The reachability issue must also be considered since the unreachability of some nodes during the execution of the group key establishment protocol might lead to the establishment of a key among a subgroup of members. As communication is reestablished the group key establishment protocol must be reinitiated or a protocol for changing the group key must be executed in order to include the previously unreachable group members and provide backward and forward secrecy.

A problem rises if during the new group key establishment one or more nodes that already had the group key become unreachable and do not get access to the new group key. The reacceptance of these nodes in the group might demand another group key change, leading to constant modification of the group key due to the temporary unreachability of some group members. This group key instability causes the communications among the group members to be inefficient and insecure.

4 Final comments

This paper introduced the issue of secure group communications. Some protocols based on the Diffie-Hellman algorithm were presented and their use in wireless ad hoc environments was analyzed.

The CLIQUES protocol suite was considered the best for ad hoc environments due to its good performance in respect to the number of messages and to the number of exponential operations required, to the fact that there is no need for synchronism among the group members when its time to send the contributions, and for allowing the establishment of the group member sequence to take place during the group key establishment.

A protocol for group members discovery was proposed. This protocol allows the establishment of a sequence among the group members that is used for group key establishment. It was shown that the unreachability issue might lead to instability of the group key. The solution for this problem remains an open field for future works.

References

[1] Zygmunt J. Haas and Lidong Zhou, "Securing ad hoc networks," *IEEE Network Magazine*, vol. 13, no. 6, pp. 24–30, Nov./Dec. 1999.

---

[5]A flooding message is retransmitted only once by every node that receives it.

[2] Michael Steiner, Gene Tsudik, and Michael Waidner, "Diffie-Hellman key distribution extended to group communication," *3rd ACM Conference on Computer and Communications Security*, Mar. 1996.

[3] Whitfield Diffie and Martin Hellman, "New directions in cryptography," *IEEE Transactions on information Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[4] Ingemar Ingemarsson, Donald T. Tang, and C. K. Wong, "A conference key distribution system," *IEEE Transactions on Information Theory*, vol. IT-28, no. 5, pp. 714–720, Sept. 1982.

[5] Mike Burmester and Yvo Desmedt, "A secure and efficient conference key distribution system," in *Advances in Cryptology – EUROCRYPT '94*, May 1994, pp. 275–286.

[6] Klaus Becker and Uta Wille, "Communication complexity of group key distribution," in *5th ACM conference on Computer and Communication Security*, Nov. 1998.

[7] Michael Steiner, Gene Tsudik, and Michael Waidner, "CLIQUES: A new approach to group key agreement," *18th International Conference on Distributed Computing Systems*, May 1998.

[8] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik, "Authenticated group key agreement and friends," in *Proceedings of the 5th ACM Conference on Computer and Communications Security*, Nov. 1998.

[9] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik, "New multiparty authentication services and key agreement protocols," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, Apr. 2000.

[10] Michael Steiner, Gene Tsudik, and Michael Waidner, "Key agreement in dynamic peer groups," *IEEE Transactions on Parallel and Distributed Systems*, vol. 11, no. 8, pp. 769–780, Aug. 2000.