24

Secure group communication: a dynamic approach

Nikos Alexandris[1], *Mike Burmester*[2] *and Vassilis Chrissikopoulos*[1]

[1] *Department of Informatics, University of Piraeus*
80 Karaoli & Dimitriou Str., 185 34 Piraeus, Greece
Phone +301 4120751, Fax +301 4112463
email alexandr@unipi.gr, chris@unipi.gr

[2] *Royal Holloway, University of London*
Department of Mathematics, Egham, Surrey TW20 OEX, U.K.
Phone +44 1784 443084, Fax +44 1784 4430766
email m.burmester@rhbnc.ac.uk

Abstract

Key distribution is a major cryptographic component for secure communication. To ensure privacy data must be encrypted with keys which are distributed securely. The keys must be properly authenticated. Most of the research on key distribution has focussed on two-party systems although there is some notable work on conference systems (with more than two parties). However the dynamic aspect of such systems has been neglected. In this paper we address this issue and consider a scenario appropriate for internet applications. We show how a conference system can be extended efficiently to enable new participants to join, in such a way that all earlier communication within the conference is protected. We also consider the case when participants leave a conference. Our approach is general and can be used with any conference system. The security of our dynamic extension is essentially the same as that of the conference system. Finally we discuss several threats to dynamic conferencing.

Keywords

Key distribution, conference systems, multi-party key agreement, authentication, data security

1 INTRODUCTION

Data security in computer networks is becoming increasingly important due to the extensive use of distributed computation and databases, telecommunication applications, and more recently, internet applications. For message

Communications and Multimedia Security Vol.3 S. Katsikas (Ed.)
©IFIP 1997 Published by Chapman & Hall

confidentiality and integrity data must be encrypted with keys which are securely distributed. To prevent impersonation the keys must be authenticated [8]. Research has concentrated mainly on the design of two-party key distribution (as in [9, 16, 15, 14, 17, 20, 3, 10, 1, 19]), although there is some notable work on conference systems (e.g. [12, 13, 5, 7]). However the dynamic aspect of such systems has been neglected. In this paper we address this issue and consider several scenarios. We do not necessarily restrict the power of the adversary or require on-line trusted third parties, except where required by the mechanisms used in the implementations.

We propose an efficient technique which uses a (any) conference system as a primitive and makes it possible for new participants to join the conference and obtain a key in such a way that all earlier communication within the conference is protected (conference extension). The cost is essentially proportional to the number of new users. We also consider the case when participants leave a conference (conference restriction). We show that if the conference system used is optimal, then there is no algorithm for constructing and distributing securely a new key to a subconference, which is more efficient than the original system.

Conference systems are subject to various security threats. These may involve *man-in-the-middle* attacks* [2], *interleaving* attacks* [3, 1], and *insider* attacks. An insider is a legitimate participant (with valid secret keys) who does not adhere strictly to the conference protocol. Insiders may create dummy sessions and try to use the messages which are exchanged during the distribution of the keys to obtain information about previous session keys [4]. They may also collude to prevent certain participants from joining a conference [11].

Distributed conference keys must be 'fresh', that is, for each conference session a new key must be distributed. Furthermore, when a conference is extended, the new participants should not be able to compute the old conference key. Similarly when a conference is restricted, those participants in the old conference who are not in the new conference should not be able to compute the new key. These issues, and other security issues are discussed in more detail in Section 2 and Section 4.

The organization of this paper is as follows. In Section 2 we describe our technique for extending, merging, and restricting conference systems. In Section 3 we consider implementations and describe a particular application. In Section 4 we discuss the security of dynamic conferencing. We conclude with remarks.

---

*The adversary diverts the messages which are exchanged by the participants to a conspirator.

*A man-in-the-middle attack in which the adversary interleaves messages which are exchanged by the participants in *different* sessions.

2 DYNAMIC CONFERENCES

A conference key distribution system is a process by which a session key is generated and distributed among m parties, $m \geq 2$. We call these parties, the *participants* of the conference, and the session key the *conference* key. The conference key is used to encrypt data communicated within the conference, for confidentiality and integrity. To prevent impersonation attacks, the conference key must be authenticated by the participants.

Several conference systems have been proposed in the literature. In this paper we are not going to be restricted to a particular system. Any conference system will do. We shall refer to it as the *primitive* system. This system will be extended to get a dynamic system. Of course the security of the dynamic system can be no better than that of the primitive system used (in fact in our case it will turn out to be roughly the same).

In our model we assume that all pairs of parties share a 'long-life' secret key. This key is obtained either directly from a Trusted Center, or indirectly from secret information (e.g. based on public keys) which has been certified by the Trusted Center (in Section 3 we discuss two implementations). As opposed to the long-life keys, conference keys should be used for only one session. That is, with each new session a 'fresh' key should be used. There are many reasons for this. One is to prevent ciphertext attacks. Other reasons are key compromise, replay attacks, and preventing information being carried across distinct sessions.

In our scheme, the messages exchanged by the participants in a conference are encrypted using the conference key K. For this purpose we use an appropriate symmetric key encryption scheme $\mathcal{E}_K$. We do not specify this scheme any further. However we shall assume that it is at least as strong as the conference primitive.*

2.1 Extending a conference

New participants may wish periodically to join a conference. If the same key K is used for encryption, then the new participants will have access to all earlier conference messages encrypted under K. This will expose the system to several security threats, and is not desirable. So a new key K' has to be constructed and distributed. There are several ways in which this can be achieved. In this paper we propose techniques which use the original conference key exchange protocol as a primitive. Our goal is to minimize the complexity of computing a new key.

2.1.1 Adding new members. Let $\mathcal{U} = \{U_1, \ldots, U_n\}$ be a conference with session key K and let $U'_1, U'_2, \ldots, U'_t$ be the parties who wish to join the

*DES in Cipher Block Chaining mode is satisfactory for most applications.

conference. One member of $\mathcal{U}$, say U_n, generates a conference K' with the new participants $U'_1, U'_2, \ldots, U'_t$ by using the conference primitive. The key K' is then concatenated with a list consisting of the labels of U_n and all new participants, and sent to the participants in $\mathcal{U}$ encrypted under the old key K (Figure 1).

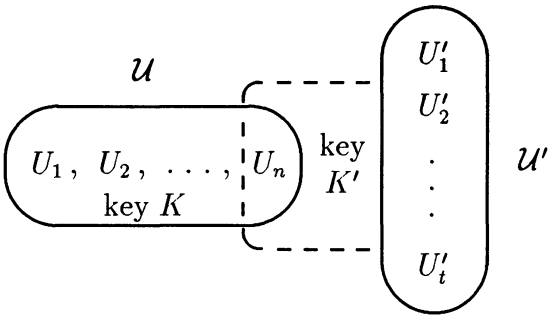

Figure 1. Adding new members

With this technique the cost of extending a conference of n participants to allow for t new participants is minimized, and is roughly proportional to $t+1$ and not $t+n$. More specifically, if $\Phi(n)$ is the overall complexity* of distributing a key to n parties using the primitive, then the complexity of our extension is $\Phi(t+1)+e$, where e is the complexity of encrypting and sending the new key. For one new participant ($t=1$) this is $\Phi(2)+e$, which clearly cannot be reduced significantly,* unless a better primitive is used. In general, if an optimal conference system is used then the complexity of extending it to include t new participants cannot be less than $\Phi(t+1)$, otherwise this system could be used to compute a conference key for $t+1$ participants with complexity less than the optimal.

Sequential extensions. New members can be added sequentially. For this purpose, the new parties are arranged into subgroups $\mathcal{U}'_{i_k}$, $k=1,\ldots,t, t \leq n$, each one of which is subordinate to a member U_{i_k}, $k=1,\ldots,t$, in $\mathcal{U}$. These subgroups then generate conference keys K'_{i_k} using the conference primitive. The keys K'_{i_k} are sent by U_{i_k} to all the members in $\mathcal{U} \cup (\cup_{j=i_1}^{j=i_k} \mathcal{U}'_j)$ encrypted as above.

2.1.2 Merging groups. Another way of extending conferences is by merging.

*This includes the computational and communication complexity of the multi-party computation.

*At least two parties must exchange a key.

If the groups are $\mathcal{U},\mathcal{U}'$ with keys K, K', and $\mathcal{U} \cap \mathcal{U}' = \emptyset$, then two participants U, U', one in each group, exchange a key K^* using the primitive. This key is then sent to all parties in $\mathcal{U}$ encrypted under K by U, and to all parties in $\mathcal{U}'$ encrypted under K' by U' (see Figure 2). If the groups overlap, then a participant $U \in \mathcal{U} \cap \mathcal{U}'$ chooses a random key K^* in the keyspace and sends it to all parties in $\mathcal{U},\mathcal{U}'$ encrypted under the appropriate key (K or K').

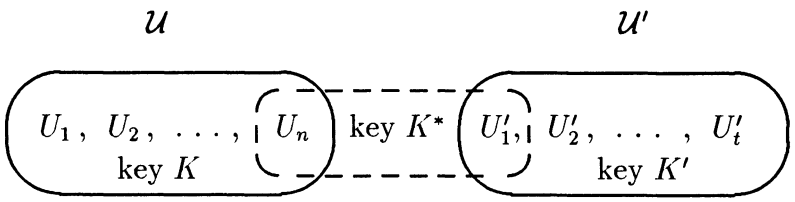

Figure 2. Merging groups

2.2 Restricting a conference

When some participants leave the conference then the session key K has to be changed, otherwise these parties will have access to all future messages encrypted under the key K. If the parties which leave have joined at an earlier stage using the key extension protocol described in Section 2.1 then the remaining participants can revert to the old session key. Otherwise a new session key must be generated and distributed. In this case, the best procedure to distribute a new key is to use the conference primitive directly. To show this we prove the following:

Proposition. *If the primitive system is optimal then there is no algorithm which can distribute securely a new key to a restricted conference with complexity less than the primitive.*

Proof. Suppose that t parties leave a conference of n participants ($n > t$) and that a new (fresh) key has to be computed. Let $\Phi(n)$ be the complexity of an optimal conference system **A** for n participants, and let **A'** be an algorithm which on input the (old) conference key(s) K and the secret keys of the $n-t$ participants (or their shared keys), outputs with complexity less than $\Phi(n-t)$ a session key K'. If **A'** is secure then it should be computationally infeasible* to compute K' from the key K and the secret keys (or the shared secret keys) of the remaining t parties who leave the conference. Therefore algorithm **A'** can be used as a conference system for $n-t$ users with complexity less than that of **A** (for $n-t$ users). As input for **A'** we select any key K for the old

*Or information theoretically impossible, depending on the model used.

conference key* and any keys for the remaining $n - t$ participants (the keys should be selected in appropriate fields). This contradicts our assumption that algorithm **A** was optimal.

2.3 Rekeying

In many applications it is desirable that conference sessions are periodical rekeyed. One reason is to prevent the key being compromised and to prevent ciphertext attacks. Another is to support various conferencing services such as session registration for new participants [18]. The technique proposed in Section 2.1 can easily be used for this purpose.

3 IMPLEMENTATIONS

We consider two methods for implementing dynamic conference systems. The first is based on probabilistic encryption and uses pseudo-random functions. All pairs of participants are given privately long-lived common secret keys by a Trusted Center, which are used as keys for pseudo-random functions. These are then used to define a conference system as in [1, 6]. This method is essentially provably secure, provided the adversary is polynomially bounded (and cannot distinguish pseudo-random sequences from random sequences). Suitable pseudo-random functions for this method of implementation can be obtained by using the construction proposed by Bellare and Rogaway. This uses the Data Encryption Standard (DES) in Cipher-Block-Chain mode and a suitable hash function.

The second method is based on the Diffie-Hellman key exchange [9]. A Trusted Center publishes an appropriate prime p and a primitive element g (or an element whose order is a large factor of $p-1$). Each party U_i selects its own secret key s_i, which is a random number in Z_{p-1}, and registers $P_i = g^{s_i} \bmod p$ with the Trusted Center. The long-life keys are then computed using the Diffie-Hellman key exchange, and used with a conference system based on the Diffie-Hellman scheme [5, 6]. Breaking this system is as hard as breaking the Diffie-Hellman problem (provided an appropriate encryption scheme is used).

3.1 A particular application based on a public-key conference system

As primitive we use the cyclic conference system proposed in [7, 5] which is an extension of the Diffie-Hellman key exchange [9]. For this, each user in $\mathcal{U} = \{U_1, \ldots, U_n\}$ has a secret key $s_i \in Z_{p-1}$ and a public key $P_i =$

*We discount the cost of simulating old keys.

$g^{s_i} \bmod p$. To compute a conference key each user U_i sends to U_{i+1} initially $X_i = g^{r_i} \bmod p$, where r_i is a random exponent in Z_{p-1}, and then $Y_i = (P_{i+1})^{r_i}/(X_{i-1})^{s_i} \bmod p$. The conference key is $K = g^{r_1 s_2 + r_2 s_3 + \ldots r_n s_1} \bmod p$, which all parties can compute.

Suppose that two new participants U_1', U_2' wish to join the conference $\mathcal{U}$. We employ the technique described in Section 2. Thus U_n, U_1', U_2' compute the key $K = g^{r_n s_1' + r_1' s_2' + r_2' s_n} \bmod p$, using the conference system described above. The other parties $U_1, U_2, \ldots, U_{n-1}$ do not participate, but receive the key K' encrypted under their key K by U_n. The overall computational cost for this extension is only $3 \times 3 = 9$ exponentiations, roughly, compared to $(n+2) \times 3$ exponentiations otherwise.

4 SECURITY ASPECTS OF DYNAMIC CONFERENCE SYSTEMS

The dynamic conference system discussed above is, essentially, as secure as the underlying primitive. In Section 1 and Section 2 we have discussed various security threats to conference systems. In this section we focus on those threats which relate to dynamic conferencing.

It is usually assumed that insiders (with valid secret keys) in a cryptographic system are honest, i.e. behave according to the protocol. However, as pointed out earlier this is not necessary always the case. Indeed some of the more serious threats come from dishonest insiders. One such threat is *exclusion*. With this, a group of insiders arranges that some participants will not receive the conference key, but that all the other participants will believe that these parties are in the conference. For example, in the dynamic extension in Figure 1, participant U_1 may exchange a conference key K' with only $U_2', \ldots, U_t'$, but then include U_1' in the encryption $\mathcal{E}_K(K', U_1, U_1', \ldots, U_t')$, thus claiming that U_1' is also a new member. In this case, all the participants in $\mathcal{U}$ will believe that U_1' is in the extended conference, whereas U_1' has not receive the key K'. Similarly, some of the new participants may be *shielded* by dishonest insiders as in [11] (if the primitive conference system allows for this). To prevent these types of attack, all the new participants must sign: an encryption under the key K' of a list consisting of the new participants and the random strings sent in the conference key exchange, using public keys. This encryption must be sent to all the other participants in the extended conference. In the symmetric key case all new participants must send to all the other participants in the extended conference an encryption under the corresponding shared key of the list of the new participants and the random strings exchanged.

Another possible attack is to prevent some of the parties from receiving the conference key, or some encrypted messages, by *blocking* paths of the physical communication network. This can be prevented by using a communication network with sufficient connectivity.

The security of our system is, essentially, based on the transitivity of authentication. More specifically, if two authenticated groups $\mathcal{U}$,$\mathcal{U}'$ intersect, then an authenticated key for $\mathcal{U} \cup \mathcal{U}'$ can be obtained by simply sending an encrypted key*. If on the other hand $\mathcal{U} \cap \mathcal{U}' = \emptyset$, then some parties in $\mathcal{U}$ (e.g. U_n) must establish an authenticated key with some members in $\mathcal{U}'$ (e.g. U_1'). This key is then used as the authenticated key for $\mathcal{U} \cup \mathcal{U}'$. Transitivity of authentication guarantees that the new key is authenticated.

5 CONCLUDING REMARKS

We have described a dynamic approach for secure group communication. Our technique for extending conferences has minimal cost. We have also shown that the most efficient way to restrict a conference is to use the primitive directly.

The security of the dynamic approach is no worse than that of the primitive. We get a secure dynamic system if a strong primitive system is combined with a good encryption scheme.

REFERENCES

[1] Bellare, M. and Rogaway, P. (1994) Entity authentication and key distribution, in *Advances in Cryptology, Crypto '93, Lecture Notes in Computer Science 773* (ed. D.R. Stinson), Springer-Verlag, 232–249.

[2] Bengio, S., Brassard, G., Desmedt, Y.G., Goutier C. and Quisquater, J.-J. (1991) Secure implementations of identification systems. *Journal of Cryptology*, **4**(3), 175–183.

[3] Bird, R., Gopal, I., Herzberg, A., Jansen, P., Kutten, S., Molva, S.R. and Yung, M. (1992) Systematic design of two-party authentication protocols, in *Advances in Cryptology, Crypto '91, Lecture Notes in Computer Science 576* (ed. J. Feigenbaum) Springer-Verlag, 44–61.

[4] Burmester, M. (1994) On the Risk of Opening Distributed Keysn in *Advances in Cryptology, Crypto '94, Lecture Notes in Computer Science 839* (ed. Y. Desmedt) Springer-Verlag, 308–317.

[5] Burmester, M. and Desmedt, Y. (1995) A Secure and Efficient Conference Key Distribution System, in *Advances in Cryptology, Eurocrypt '94, Lecture Notes in Computer Science 950* (ed. A. De Santis) Springer-Verlag, 275–286.

[6] Burmester, M. and Desmedt, Y. (1996) Efficient and Secure Conference Key Distribution. *Proceedings of the International Workshop on Se-*

*As pointed out earlier, to prevent exclusions, when these are of consequence, the new group must be confirmed.

curity Protocols, Lecture Notes in Computer Science 1189 (ed. M. Lomas) Cambridge, 10-12 April 1996, 119–129.

[7] Chrissikopoulos, V. and Peppes, D. (1995) A Practical Conference Key Distribution System, in *Information Security – the Next Decade, Proc. IFIP/SEC95* (eds. J. Eloff and S. Solms) Chapman and Hall, 168-175.

[8] Denning, D.E.R. (1982) Cryptography and Data Security. Addison-Wesley, Reading, MA.

[9] Diffie, W. and Hellman, M.E. (1976) New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22(6), 644–654.

[10] Diffie, W., van Oorschot, P.C. and Wiener M.J. (1992) Authentication and authenticated key exchanges. *Designs, Codes and Cryptography*, **2**,107–125.

[11] Justin, M. and Vaudeney, S. (1996) Authenticated Multi-party Key Agreement, in *Advances in Cryptology, Asiacrypt '96, Lecture Notes in Computer Science 1163* (eds. K. Kim and T. Matsumoto), Springer-Verlag, 36–49.

[12] Ingemarsson, I., Tang, D.T. and Wong, C.K. (1982) A conference key distribution system. *IEEE Trans. Inform. Theory*, **28**(5), 714–720.

[13] Koyama, K. and Ohta, K. (1988) Identity-based conference key distribution systems, in *Advances in Cryptology, Proc. of Crypto '87, Lecture Notes in Computer Science 293* (ed. C. Pomerance) Springer-Verlag, 175–185.

[14] Kohl, J., Newmann, B.C. and Ts'o, T. (1994) The evolution of the Kerberos Authentication System. *Distributed Open Systems*, IEEE Computer Society Press, 78–94.

[15] Matsumoto, T. , Takashima Y. and Imai, H. (1986) On seeking smart public-key distribution systems. *Transactions of the IECE*, **69**, 99–106.

[16] Needham, R.M. and Schroeder, M.D. (1978) Using Encryption for Authentication in Large Networks of Computers. Commun. ACM, **21**, 993–999.

[17] Okamoto, E. (1988) Key distribution systems based on identification information, in *Advances in Cryptology, Proc. of Crypto '87, Lecture Notes in Computer Science 293* (ed. C. Pomerance) Springer-Verlag, 194–20.

[18] Oppliger, R. and Albanese, A. (1996) Distributed registration and key Distribution (DiRK), in *Information Systems Security, Facing the information Society of the 21st century, IFIP SEC '96.* (eds. S.K. Katsikas, D. Gritzalis) Chapman & Hall, 199–208.

[19] Rueppel, R.A. and van Oorschot, P.C. (1994) Modern key agreement techniques. *Computer Communications*, **17**, 458-465.

[20] Yacobi, Y. and Shmuely Z. (1990) On key distribution systems, in *Advances in Cryptology, Crypto '89, Proceedings, Lecture Notes in Computer Science 435* (ed. G. Brassard), Springer-Verlag, 344–355.

6 BIOGRAPHY

Nikos Alexandris is a Professor in the Department of Informatics at the University of Piraeus. He received his BSc from Athens University and his PhD from UMIST in 1978. His research interests include Expert Systems, Information Security and Cryptography. He is a member of the Greek Mathematical Society, the Greek Computer Society and the BCS.

Mike Burmester is in the Information Security Group at Royal Holloway, University of London and is a Reader in Mathematics. He received his BSc from Athens University and his PhD from the University of Rome. Since 1990 he has been working in Cryptography and Computer Security. He is a member of the Greek Mathematical Society, the International Association of Cryptological Research and a Fellow of the Institute of Mathematics and its Applications.

Vassilis Chrissikopoulos is an Associate Professor in the Department of Informatics at the University of Piraeus. He received his BSc from the University of Thessaloniki and his PhD from Royal Holloway, University of London. His research interests include Expert Systems, Information Security and Cryptography. He is a member of the Greek Mathematical Society, the Greek Computer Society, Operational Research Society in the UK and the BCS.