Comments on two anonymous conference key distribution systems

Qiang Tang and Chris J. Mitchell
Information Security Group
Royal Holloway, University of London

30th August 2004

Abstract

In this paper we consider the security of two recently proposed anonymous conference key distribution schemes. We show that neither scheme is as practical as the authors claim and that, in certain circumstances, both schemes also suffer from security vulnerabilities. We also show that the attack described in one paper is invalid.

Keywords: Key distribution, User anonymity.

1 Introduction

In [1], Yang, Chang and Hwang propose a new anonymous conference key distribution system (referred to here as the YCH scheme). The authors claim that this scheme is efficient, secure, and provides user anonymity, based on the intractability of the elliptic curve discrete logarithm problem. However, we show below that this scheme is not efficient, because every conferee must, on average, try $\frac{n+1}{2}$ times to get the session key, where n is the size of the conference. We also show that, in certain circumstances, malicious users can successfully manipulate the session key for the conference.

In [2], Lin, Lin and Lee describe an attack on the scheme in [1] and propose an improved anonymous conference key distribution system (referred to here as the LLL scheme) based on the YCH scheme. However, we show that their attack is incorrect and cannot be used to attack the YCH scheme. The new LLL scheme also suffers from the same efficiency and security problems as the YCH scheme does.

The rest of this paper is organised as follows. In Section 2, we give a concise description of the YCH scheme and the LLL scheme. In Section 3, we describe a general model for both schemes. In Section 4, we give our comments on both schemes. In Section 5, a brief conclusion is provided.

2 Description of the YCH and LLL schemes

Both schemes involve two kinds of entity, namely the conference chairman (U_c) and the conferees, and both schemes are composed of three stages: the system initialisation stage, the conference key distribution stage, and the conference key recovery stage. The operations in the three stages of both schemes are basically the same, so we only describe these three stages for the YCH scheme and point out the differences between them at relevant points. Suppose that the set of all participants in the system is $A = \{U_1, U_2, \cdots, U_m\}$.

- In the system initialisation stage, the system publicly chooses an elliptic curve E over a finite field $GF(q)$ and a base point $G \in E_q$, whose order is a very large number p. Then the system assigns a secret key $x_i \in [1, p-1]$, the corresponding public key $Q_i = x_i G$, and the identity ID_i to participant U_i $(1 \leq i \leq m)$.

  The operations of this stage in the LLL scheme are identical.

- In the conference key distribution stage, U_c, the conference chairman, distributes a conference key CK to each participant in the conference, which, without loss of generality, we denote by U_i $(1 \leq i \leq n, n < m)$. Note that in the description we implicitly assume that U_c is not a member of $\{U_1, U_2, \cdots, U_m\}$ — see also the comments in Section 4. However, modifying the specification for the case where U_c is a member of $\{U_1, U_2, \cdots, U_m\}$ would be straightforward.

  U_c first randomly chooses a value c_1, $0 < c_1 < p$, and then performs the following steps for each U_i $(1 \leq i \leq n)$.

  1. Compute the secret key $k_{ci} = x_c Q_i$, shared by U_c and U_i.
  2. Compute the hash value $h_i = H(k_{ci}||ID_c||ID_i||T)||m$, where H is a secure one-way hash function with fixed-length output, T is a time-stamp, and $||$ denotes the concatenation operation.

3. Compute $y_i = c_1 h_i + CK \bmod p$.

U_c computes $V = H(CK||ID_c||T)$ as a check value for CK and the time-stamp T, and then broadcasts the message $M = \{ID_c, V, T, c_1, y_1, y_2, \cdots, y_n\}$ to all the conference participants. Note that the identities of the conference participants are not included in M — this is to ensure anonymity for the conference participants. However, as we discuss in more detail below, this means that every recipient of M has to process each y_i in turn to see if the message is intended for them.

The operations of this stage in the LLL scheme are identical except that, in step 3, y_i is computed as $y_i = (c_1 h_i + CK \bmod p) \oplus h_i$, where $\oplus$ denotes bit-wise exclusive-or.

- In the conference key recovery stage, on receiving the message $M = \{ID_c, V, T, c_1, y_1, y_2, \cdots, y_n\}$ each user U_j $(1 \leq j \leq m)$ performs the following steps to determine whether or not they are a member of the conference and, if so, to recover the conference key CK. Note that U_j needs to process each value y_i $(1 \leq i \leq n)$ in turn to determine whether or not this value is intended for U_j, although the authors in [1, 2] do not point this out.

  1. Check the validity of the time-stamp T. If it is invalid, stop the conference key recovery stage.
  2. Compute the key $k_{jc} = x_j Q_c$ shared with U_c.
  3. For each i, $1 \leq i \leq n$, perform the following steps using the value y_i:
     (a) Compute $h'_i = H(k_{jc}||ID_c||ID_j||T)||m$, and use it to compute CK' as
     $$CK' = y_i - c_1 h'_i \bmod p.$$
     (b) Check the validity of CK' by computing $H(CK'||ID_c||T)$ and comparing it with V.
     (c) If they agree, then U_j knows that it is a member of the conference and that CK' is the valid secret key for the conference; U_j stops processing in this case. Otherwise U_j continues to try.
  4. If all the iterations of step 3 complete without successfully finding a valid conference key, then U_j knows that it is not a member of the conference and stops processing.

The operations of this stage in the LLL scheme are identical except that step 3(a) is as follows.

Compute $h'_i = H(k_{jc}||ID_c||ID_j||T)||m$, and compute CK' as

$$CK' = (y_i \oplus h_i) - c_1 h'_i \bmod p.$$

3 General model for the YCH and LLL schemes

Both the YCH and the LLL schemes conform to the same simple general model, as follows, where the objective is for a conference chair U_c to share a conference key CK with all the members of a conference, namely U_1, U_2,$\cdots$, U_n.

1. The conference chair U_c shares a secret key K_i with participant U_i, $1 \leq i \leq n$.
2. U_c derives a session key h_i from K_i and a time-stamp T.
3. For each participant U_i $(1 \leq i \leq n)$, U_c encrypts CK using the session key h_i, i.e. computes $e_{h_i}(CK)$, where e denotes an encryption operation.
4. The conference chair broadcasts y_1, y_2, $\cdots$, y_n, along with a time-stamp T, its identifier ID_c, and a check value $V = H(CK||ID_c||T)$.
5. Each recipient U_i attempts to decrypt all the values y_1, y_2, $\cdots$, y_n, using the secret session key h_i, and then in each case checks that the decrypted value is correct using the check value V.

In addition, both the YCH scheme and the LLL scheme require the secret keys K_i to be generated by an elliptic curve version of the Diffie-Hellman key establishment technique. However, the schemes would work in an identical way if a conventional Diffie-Hellman scheme was used to derive K_i, or even if they were pre-distributed by some means.

These two schemes differ only in their choices for the encryption operation e, which are

$$y_j = e_{h_i}(CK) = c_1 h_i + CK \bmod p$$

and

$$y_j = e_{h_i}(CK) = [c_1 h_i + CK \bmod p] \oplus h_i$$

for the YCH and LLL schemes respectively.

Note that this means that, for the YCH scheme, a genuine recipient U_i can use the recovered copy of CK to easily compute all the values h_j $(j \neq i)$, simply by computing $h_j = (y_j - CK)c_1^{-1} \bmod p$, where c_1^{-1} is the inverse of c_1 modulo p. Recovering the values h_j in the LLL scheme is rather more difficult, although not impossible, since the encryption function e used in that scheme is still not secure. To see this, observe that if the conference chair sends out two different conference keys using the same time stamp T, and if one user U_j is in both conferences, then detecting this and recovering the value h_j appears straightforward.

4 Comments on the two schemes

We first point out that the specifications of both schemes are incomplete.

1. Firstly, for neither scheme is it specified whether U_c is a member of the underlying conference and how the key pair (x_c, Q_c) is generated. As noted above, for the purposes of this paper we have assumed that U_c is not a member of $\{U_1, U_2, \cdots, U_m\}$.

2. Secondly, it is not at all clear why (in step 2 of the key distribution stage) the value of m, the total number of participants in the scheme, is appended to the hash-code to derive h_i. This does not appear to increase the security of the scheme or play any other useful role. It is also not specified how m is communicated to the participants — it could, perhaps, be conveyed at the system initialisation stage, although this would force m to be static, and prevent the addition of new participants in the scheme.

We also have the following comments on both schemes.

1. In [2], the authors claim that, using the message

   $$M = \{ID_c, V, T, c_1, y_1, y_2, \cdots, y_n\},$$

   an attacker can learn the parameters h_i $(1 \leq i \leq n)$ by first solving the n equations:

   $$y_i = c_1 h_i + CK \bmod p, \quad 1 \leq i \leq n,$$

   then eliminating CK from these equations to derive a set of n equations of the form $(y_i - y_{i+1}) = c_1(h_i - h_{i+1}) \bmod p$, and finally solving these equations to yield the n unknowns h_i, and hence the secret key CK. However, they fail to observe that the derived equations are linearly dependent, and hence cannot be uniquely solved. In fact, it should be clear that any value of CK will be consistent with these equations.

2. We argue that the computational complexity for each participant in the conference key recovery stage specified in part 5 of [1] is incorrect. The authors claim that the total computational complexity for each participant is $T_{EC\_MUL} + 2t_H + T_{MUL}$, where T_{EC_MUL}, T_H and T_{MUL} are the time for computation of elliptic curve multiplication, the hash function H, and modular multiplication, respectively. But, in the conference key recovery stage of the scheme in [1], a recipient of M cannot immediately determine which y_i $(1 \leq i \leq n)$ (if any) is for him. He must try every y_i $(1 \leq i \leq n)$ in step 3 and then check the validity of his result. The expected number of tries to successfully get the session key CK is $\frac{n+1}{2}$, which will be a heavy burden for the conferees if n is very large. The new scheme in [2] also suffers from this efficiency problem.

3. The reformulation of the scheme in general terms given in the previous section shows that neither of the proposed schemes is in any way elliptic curve specific; the descriptions of the schemes given in the two papers is thus very misleading. In fact the schemes do not even necessitate the use of any form of public key cryptography. All that is required is some means of establishing a shared secret key between the conference chair and each participant. This could be achieved using an arbitrary asymmetric key establishment scheme, or even pre-established shared secrets.

In addition, both schemes suffer from security vulnerabilities in certain circumstances. In the YCH scheme, as we have described above, a malicious but genuine participant U_i can use the conference key CK to recover the h_j 'session key' values for all other participants U_j. Participant U_i can now use these session keys to reformulate the message M to include a different conference key CK^*, by sending out modified values $y_j^* = c_1 h_j + CK^*$ and a recomputed check value V^*.

This attack is not so simple for the LLL scheme. However, as we have already noted, there are circumstances where a similar attack could apply since the encryption function e for the LLL scheme is still not secure. Such problems could be avoided by using a secure encryption function, such as AES [3], for e.

The schemes as described do not enable the recipient of a broadcast conference keying message to determine whether or not it has been tampered with; in particular, the verification value V is not a function of the set of conference members. Thus the schemes should only ever be used in environments where such tampering is not a threat, e.g. where communications are protected by other means.

5 Conclusion

In this paper, we have analysed two recently proposed anonymous conference key distribution schemes, and have demonstrated the presence of significant design vulnerabilities. We also point out that the 'attack' in [2] is invalid.

6 Acknowledgements

The authors would like to express deep appreciation to the reviewers for their valuable comments.

References

[1] C. C. Yang, T. Y. Chang, and M. S. Hwang. A new anonymous conference key distribution system based on the elliptic curve discrete logarithm problem. *Computer Standards & Interfaces*, 25:141–145, 2003.

[2] C. H. Lin, C. Y. Lin, and W. Lee. Comments on the Yang-Chang-Hwang anonymous conference key distribution system. *Computer Standards & Interfaces*, 26:171–174, 2004.

[3] J. Daemen and V. Rijmen. *The Design of Rijndael: AES — The Advanced Encryption Standard*. Springer-Verlag, 2002.