A Weakness in Jung-Paeng-Kim's ID-based Conference Key Distribution Scheme

Junghyun Nam, Seungjoo Kim, and Dongho Won

School of Information and Communication Engineering,
Sungkyunkwan University, 300 Chunchun-dong, Jangan-gu, Suwon-si,
Gyeonggi-do 440-746, Korea
jhnam@dosan.skku.ac.kr, skim@ece.skku.ac.kr, dhwon@dosan.skku.ac.kr

Abstract. Very recently, Jung, Paeng and Kim [IEEE Communications Letters, Vol 8, No 7, pp 446–448, July 2004] have demonstrated the insecurity of Xu and Tilborg's ID-based conference key distribution scheme, and in addition, have revised the scheme to fix the security flaws discovered by them. However, in this paper, we show that Jung-Paeng-Kim's revised scheme is still insecure since it is vulnerable to an active attack of colluding adversaries. We also show that our attack can be easily thwarted by a simple patch.

Keywords: Conference key distribution; active attack; colluding adversaries.

1 Introduction

Secure key distribution is of vital concern to anyone interested in communicating securely over an untrusted, open network. However, the experience has shown that the design of key establishment protocols that are secure against an active adversary is not an easy task to do, especially in a multi-party setting. In 2000, Xu and Tilborg [3] proposed an ID-based conference key distribution scheme which builds on earlier work of Harn and Yang in the 2-party setting [1]. However, four years later, Jung et al. [2] have pointed out that Xu and Tilborg's scheme does not meet forward secrecy and is vulnerable to impersonation attacks. Furthermore, they have proposed a revised version of the Xu-Tilborg scheme which appears to provide resistance against their attacks. But unfortunately, Jung et al.'s revised scheme is still insecure against an active adversary. In this paper, we identify the vulnerability of Jung et al.'s revised scheme to an attack of colluding adversaries, and present a simple patch to fix the security flaws identified in this scheme.

2 Review of Jung et al.'s Scheme

The scheme consists of three phases: the initiation phase, the user registration phase, and the application phase.

2.1 Initiation Phase

The key authentication center (KAC) selects a one-way function f, a large prime p, and a primitive element α of GF(p), which are all known to the public. Then KAC chooses as its private key a random $x \in [1, p-1]$ such that $\gcd(x, p-1) = 1$, and computes its public key y as:

$$y = \alpha^x \bmod p.$$

2.2 User Registration Phase

Let $\mathcal{P}$ denote the set of all potential users. Each user $U_i \in \mathcal{P}$ submits its identity ID_i to the KAC for registration. The KAC computes for user U_i an extended identity $\mathrm{EID}_i = f(ID_i)$ and the signature (r_i, s_i) of EID_i as

$$\begin{aligned} r_i &= \alpha^{k_i} \bmod p, \\ s_i &= (\mathrm{EID}_i - k_i r_i) x^{-1} \bmod p - 1, \end{aligned}$$

where k_i is chosen randomly from $[1, p-1]$ such that $\gcd(s_i, p-1) = 1$. Here, no k_i should be used repeatedly. Then, in the application phase, s_i and r_i will be used as user U_i's private and public keys, respectively.

2.3 Application Phase

Let $\mathcal{U} = \{U_1, \ldots, U_\ell\}$ be a nonempty subset of $\mathcal{P}$ who wish to share a conference key among them. Users are arranged in a star network, with user U_1 being the chairman who plays a central role in the protocol. Now, the users in $\mathcal{U}$ perform the following four steps to generate the conference key K_c.

1. User U_1 chooses two random values $v_1, e_1 \in [1, p-1]$ such that $\gcd(v_1, p-1) = 1$ and $\gcd(e_1, p-1) = 1$. Next, user U_1 computes

$$\begin{aligned} w_1 &= y^{v_1} \bmod p, \\ n_1 &= w_1^{e_1} \bmod p, \\ \eta_1 &= (m - v_1 w_1) s_1^{-1} \bmod p - 1, \end{aligned}$$

   where $m = f(n_1 \| \mathrm{ID}_1 \| \mathrm{time})$. Then, user U_1 sends (ID_1, r_1, w_1, n_1, η_1, time) to the rest of the users.
2. Upon receiving (ID_1, r_1, w_1, n_1, η_1, time), each user $U_i \in \mathcal{U} \setminus \{U_1\}$ verifies that the following congruence holds:

$$y^m \equiv w_1^{w_1} (\alpha^{\mathrm{EID}_1} r_1^{-r_1})^{\eta_1} \pmod p. \tag{1}$$

   If the verification succeeds, user U_i chooses two random values $v_i, e_i \in [1, p-1]$, such that $\gcd(v_i, p-1) = 1$ and $\gcd(e_i, p-1) = 1$, and computes

$$\begin{aligned} w_i &= y^{v_i} \bmod p, \\ n_i &= w_1^{e_i} \bmod p, \\ \eta_i &= (f(n_i) - v_i w_i) s_i^{-1} \bmod p - 1. \end{aligned}$$

   Then user U_i sends (ID_i, r_i, w_i, n_i, η_i) to user U_1.
3. For $i \in [2, \ell]$, user U_1 verifies that the following congruence holds:

$$y^{f(n_i)} \equiv w_i^{w_i} (\alpha^{\mathrm{EID}_i} r_i^{-r_i})^{\eta_i} \pmod p. \tag{2}$$

   If the verifications are successful, user U_1 computes the conference key K_c as

$$K_c = n_1^{e_1} \bmod p.$$

Then, for $i \in [2, \ell]$, user U_1 also computes

$$z_i = K_c \cdot n_i^{e_1} \bmod p$$

and sends $(z_i, E_{K_c}(\mathrm{ID}_1))$ to user U_i, where $E_{K_c}(\mathrm{ID}_1)$ denotes the ciphertext of ID_1 encrypted using some secure symmetric cryptosystem under the key K_c.

4. After receiving $(z_i, E_{K_c}(\mathrm{ID}_1))$, each user $U_i \neq U_1$ computes the conference key K_c as

$$K_c = z_i \cdot (n_1^{e_i})^{-1} \bmod p,$$

and verifies it through decryption of $E_{K_c}(\mathrm{ID}_1)$.

Remark: We note that two conditions $\gcd(e_1, p-1) = 1$ and $\gcd(e_i, p-1) = 1$, in steps (1) and (2) respectively, are not necessary.

3 Attack on Jung et al.'s Scheme

The fundamental security requirement for a key establishment protocol to achieve is that each protocol participant must be assured that no one aside from the intended parties can learn the value of the session key. This requirement has to be met in the presence of active adversaries who can delay, insert or delete messages. In this section we show that Jung et al.'s scheme does not satisfy this requirement.

Let A_1 and A_2 denote two colluding adversaries, and let S and S' denote two concurrent sessions of the protocol. Moreover, let $\mathcal{U}$ and $\mathcal{U}'$ be two sets of users participating, respectively, in S and S'. We assume that the adversaries A_1 and A_2 participate only in session S (i.e., $A_1, A_2 \in \mathcal{U} \setminus \mathcal{U}'$) and user U_1 plays the role of the chairman in both sessions. That is, for example, we consider the following two sessions:

$$\begin{array}{ll} \text{Session } S: & \mathcal{U} = \{U_1, A_1, A_2, U_i, \ldots, U_\ell\}, \\ \text{Session } S': & \mathcal{U}' = \{U_1, U_2, \ldots, U_{\ell'}\}. \end{array}$$

In the attack below, the goal of the adversaries A_1 and A_2 is to share a same conference key with some user, say U_2, of session S', while deluding the victim (user U_2) into believing that he has established a secure session with the users in $\mathcal{U}'$. To this end, the adversaries sit in between U_1 and U_2, facing U_1 with their true identities but facing U_2 by masquerading as user U_1. The detailed attack scenario is as follows:

1. The adversaries launch the attack by replacing the first message going to the victim (i.e., the first message sent to U_2 by U_1 for session S') with the first message

$$(\mathrm{ID}_1, r_1, w_1, n_1, \eta_1, \mathrm{time})$$

received from U_1 for session S.

2. Since the congruence (1) holds for (ID_1, r_1, w_1, n_1, η_1, time), user U_2 will believe that this message is sent for session S' as long as it arrives before the timer expires. Hence, user U_2 will compute w_2', n_2', and η_2' as per protocol specification, and will send

$$(\mathrm{ID}_2, r_2, w_2', n_2', \eta_2')$$

to user U_1. But, this message is eavesdropped on by the adversary A_2.

3. Now, using n'_2, the adversary A_2 constructs her own message to be sent to U_1 for session S; A_2 first chooses a random value $v_{A_2} \in [1, p-1]$ such that $\gcd(v_{A_2}, p-1) = 1$, and computes

$$w_{A_2} = y^{v_{A_2}} \bmod p,$$
$$\eta_{A_2} = (f(n'_2) - v_{A_2} w_{A_2}) s_{A_2}^{-1} \bmod p - 1.$$

   A_2 then sends the message $(\mathrm{ID}_{A_2}, r_{A_2}, w_{A_2}, n'_2, \eta_{A_2})$ to user U_1. Note that this message has been constructed without knowing e'_2, but is still valid. Meanwhile, the adversary A_1 behaves exactly as specified in the protocol, computing w_{A_1}, n_{A_1}, η_{A_1} and sending to U_1 the message $(\mathrm{ID}_{A_1}, r_{A_1}, w_{A_1}, n_{A_1}, \eta_{A_1})$.
4. Since the congruence (2) holds for $(\mathrm{ID}_{A_2}, r_{A_2}, w_{A_2}, n'_2, \eta_{A_2})$ and for $(\mathrm{ID}_{A_1}, r_{A_1}, w_{A_1}, n_{A_1}, \eta_{A_1})$, user U_1 proceeds to respond to the adversaries' messages; it computes

$$z_{A_1} = K_c \cdot n_{A_1}^{e_1} \bmod p \quad \text{and} \quad z_{A_2} = K_c \cdot n'^{e_1}_2 \bmod p,$$

   and sends $(z_{A_1}, E_{K_c}(\mathrm{ID}_1))$ and $(z_{A_2}, E_{K_c}(\mathrm{ID}_1))$ to the adversaries.
5. After receiving z_{A_1} and z_{A_2}, the adversaries can easily compute $n'^{e_1}_2$ even without knowing the random secret values e_1 and e'_2; they first compute the conference key K_c for session S as

$$K_c = K_c \cdot n_{A_1}^{e_1} \cdot (n_1^{e_{A_1}})^{-1} = z_{A_1} \cdot (n_1^{e_{A_1}})^{-1} \bmod p,$$

   and simply recover $n'^{e_1}_2$ as

$$n'^{e_1}_2 = K_c^{-1} \cdot K_c \cdot n'^{e_1}_2 = K_c^{-1} \cdot z_{A_2} \bmod p.$$

   With this information $n'^{e_1}_2$, the adversaries proceed to forge a message to be sent to user U_2; they first choose an arbitrary conference key K''_c (expecting U_1 to calculate K''_c as its conference key) and then compute $E_{K''_c}(\mathrm{ID}_1)$ and

$$z''_2 = K''_c \cdot n'^{e_1}_2 \bmod p.$$

6. Now, the adversaries send to U_2 the forged message $(z''_2, E_{K''_c}(\mathrm{ID}_1))$, while deleting the honest message $(z'_2, E_{K'_c}(\mathrm{ID}_1))$ sent to U_2 by U_1 for session S'. Since U_2 will think that the message $(z''_2, E_{K''_c}(\mathrm{ID}_1))$ is the response of U_1 for session S', it will compute its conference key as:

$$K''_c = K''_c \cdot n'^{e_1}_2 \cdot (n_1^{e'_2})^{-1} = z''_2 \cdot (n_1^{e'_2})^{-1} \bmod p,$$

   which is chosen by the adversaries themselves.

At the end of this scenario, the user U_2 believes that it has established a secure session with the other users in $\mathcal{U}'$ sharing a secret key K''_c, while in fact it has shared the key with the adversaries. Meanwhile, the other users in $\mathcal{U}'$ will compute the conference key K'_c as per protocol specification, and think that the session is finished successfully. As a result, the adversaries can not only access any privileged information sent from U_2 to the other users in $\mathcal{U}'$, but can also send to U_2 arbitrary messages for their own benefit pretending to be any user in $\mathcal{U}'$ (while blocking messages sent by other users in $\mathcal{U}'$ from reaching U_2).

It is straightforward to see that the attack scenario above can be easily extended so that n colluding adversaries participating in session S can share a same conference key with $n-1$ users of session S'.

4 Modification

The weakness of Jung et al.'s scheme against the attack above stems from the fact that the first message sent by U_1 in one protocol execution can be replayed in another concurrent execution even with a different set of participants. Note that integrating a timestamp into the first message of the protocol does not prevent messages from being replayed between two concurrent sessions, even if protocol participants tightly synchronize their time clocks to a global standard time such as Greenwich Mean Time. This allows the adversaries to use their true identities in one of two sessions and to successfully masquerade as the chairman in the other session. Our simple patch to this security flaw is to modify the computation of m and the first message of the protocol, respectively, to:

$$m = f(n_1 \| \mathrm{ID}_1 \| \ldots \| \mathrm{ID}_\ell) \quad \text{and} \quad (\mathrm{ID}_1, \ldots, \mathrm{ID}_\ell, r_1, w_1, n_1, \eta_1).$$

With this modification, the messages transmitted in each protocol session become bounded to the identities of all the participants of that session. It is easy to see that this is enough to protect the protocol against such an attack presented above.

However, with this modification alone, the scheme is still insecure in terms of semantic security since the conference key K_c is used as the encryption key in constructing the "authenticator" $Auth_{U_1} = E_{K_c}(\mathrm{ID}_1)$. This leaks some information about the conference key — the ciphertext of the known message ID_1 encrypted using some known algorithm under the key K_c — which allows an adversary to distinguish K_c from a random key chosen from the conference-key space. The well-known approach to avoid this common error is to compute the authenticator $Auth_{U_1}$ as

$$Auth_{U_1} = H(K_c, 1)$$

and to establish a new shared conference key K as

$$K = H(K_c, 0),$$

where H is a one-way hash function.

5 Conclusion

We have shown that Jung-Paeng-Kim's ID-based conference key distribution scheme [2] is vulnerable to an attack mounted by two colluding adversaries. We have also shown that the security hole can be easily patched by integrating users' identities into the first message of the scheme. Our work highlights again the necessity that active adversaries are to be considered carefully in designing a key establishment protocol, especially in a group setting.

References

1. L. Harn, S. Yang, ID-based cryptographic schemes for user identification, digital signature, and key distribution, IEEE Journal on Selected Areas in Communications 11 (1993) 757–760.
2. B. Jung, S. Paeng, D. Kim, Attacks to Xu-Tilborg's conference key distribution scheme, IEEE Communications Letters 8 (2004) 446–448.
3. S. Xu, H. Tilborg, A new identity-based conference key distribution scheme, in: Proc. of IEEE International Symposium on Information Theory, 2000, 269.