Computers and Electrical Engineering 36 (2010) 155–159

Contents lists available at ScienceDirect

Computers and Electrical Engineering

journal homepage: www.elsevier.com/locate/compeleceng

Enhancements of authenticated multiple key exchange protocol based on bilinear pairings

Duc-Liem Vo [a,*], Hyunrok Lee [a], Chan-Yeob Yeun [b], Kwangjo Kim [a]

[a] Korea Advanced Institute of Science and Technology, IT Convergence Campus (KAIST-ICC), Daejeon 305-732, Republic of Korea
[b] Khalifa University of Science, Technology & Research (KUSTAR), P.O.Box 573, Sharjah, United Arab Emirates

ARTICLE INFO

Article history:
Received 4 August 2008
Received in revised form 17 August 2009
Accepted 20 August 2009
Available online 25 September 2009

Keywords:
Cryptography
Authenticated key exchange
Cryptanalysis
Bilinear pairing

ABSTRACT

Lee et al. [4] proposed two new authenticated multiple key exchange protocols based on Elliptic Curve Cryptography (ECC) and bilinear pairings. In this paper, we show an impersonation attack on their pairing-based authenticated key exchange protocol. We demonstrate that any attacker can impersonate an entity to share multiple session keys with another entity of his/her choice by using only the public key of the victim. Moreover, their protocol fails to provide perfect forward secrecy, despite of their claim to the contrary. Thus, we propose a simple modification to the original protocol which avoids our attack.

1. Introduction

A key exchange protocol allows two entities to share a key which can be used to provide secure communication between them. Additionally, the key exchange protocol should provide an authentication mechanism in order to ensure that the key is only shared with between two entities. An authenticated key exchange protocol plays an important role in many modern network-based applications such as collaborative or distributed applications. In 2001, Harn and Lin [1] proposed an authentication key exchange protocol in which two parties generate four shared keys at a time, however, only three of these keys can provide perfect forward secrecy. Later, Hwang et al. [2] proposed an efficient authentication key exchange protocol requiring less computation than Harn and Lin's scheme [1]. Nevertheless, the scheme [2] was broken by Lee and Wu [3] by the modification attack. Recently, Lee et al. [4] proposed two authenticated multiple key exchange protocols: one is based on ECC and the other is based on bilinear pairings. These protocols let two entities share not only one but also four session keys in authenticated manner.

However, in this paper, we demonstrate an impersonation attack on Lee et al.'s bilinear pairing-based authenticated key exchange protocol. We also show that, using a long-term public key of an entity only, any attacker can impersonate the entity to agree some session keys with another entity. Consequently, Lee et al.'s protocol fails to provide authenticity as they have claimed. Furthermore, we indicate that perfect forward secrecy of their protocol is not guaranteed. Thus, we proposed a simple modification to the protocol which can withstand our attack.

2. Lee et al.'s authenticated multiple key exchange protocol based on bilinear pairings

We firstly review Lee et al.'s authentication multiple key exchange protocol based on bilinear pairings using the same notation as [4].

* Corresponding author.
E-mail addresses: vdliem@icu.ac.kr (D.-L. Vo), tanker@kaist.ac.kr (H. Lee), cyeun@kustar.ac.ae (C.-Y. Yeun), kkj@kaist.ac.kr (K. Kim).

0045-7906/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.compeleceng.2009.08.001

Let P be a generator of a cyclic additive group G_1 of the prime order q, and G_2 be a cyclic multiplicative group of the same order q. G_1 and G_2 are Gap Diffie–Hellman groups and the bilinear pairing is to show as $e : G_1 \times G_1 \to G_2$. Assume that Alice and Bob want to share some session keys. Let $X_A \in Z_q^*$ and $Y_A(= X_A P)$ be Alice's long-term private key and long-term public key, respectively. $X_B \in Z_q^*$ and $Y_B(= X_B P)$ are Bob's long-term private key and long-term public key, respectively. Alice and Bob execute the following steps.

Step 1. Alice → Bob: $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$.
Alice first selects two random integers, a_1 and a_2, and computes $T_{A1} = a_1 P$ and $T_{A2} = a_2 P$, where $a_1, a_2 \in Z_q^*$. Let K_{A1} and K_{A2} be the x-coordinate values of T_{A1} and T_{A2}, respectively. Then, Alice computes S_A as follows:

$$S_A = (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + X_A T_{A2}.$$

Alice sends the message $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$ to Bob, where $Cert(Y_A)$ is the certificate of Alice's long-term public key signed by a trusted party.

Step 2. Bob → Alice: $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$.
Similarly, Bob randomly selects two integers, b_1 and b_2, and computes $T_{B1} = b_1 P$ and $T_{B2} = b_2 P$, where $b_1, b_2 \in Z_q^*$. Let K_{B1} and K_{B2} be the x-coordinate values of T_{B1} and T_{B2}, respectively. Then, Bob computes S_B as follows:

$$S_B = (b_1 K_{B1} + b_2 K_{B2}) T_{B1} + X_B T_{B2}.$$

Bob sends the message $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$ to Alice, where $Cert(Y_B)$ is the certificate of Bob's long-term public key signed by a trusted party.

Step 3. Alice computes shared session keys K_1, K_2, K_3, and K_4.
Upon receiving $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$, Alice takes out the x-coordinate values K_{B1} and K_{B2} from T_{B1} and T_{B2}, separately. Then, Alice verifies:

$$e(S_B, P) \stackrel{?}{=} e(K_{B1} T_{B1} + K_{B2} T_{B2}, T_{B1}) e(T_{B2}, Y_B).$$

If this holds, Alice computes four shared session keys as follows:

$$K_1 = e(a_1 T_{B1}, Y_A + Y_B),$$
$$K_2 = e(a_1 T_{B2}, Y_A + Y_B),$$
$$K_3 = e(a_2 T_{B1}, Y_A + Y_B),$$
$$K_4 = e(a_2 T_{B2}, Y_A + Y_B).$$

Step 4. Bob computes shared session keys K_1, K_2, K_3, and K_4.
Upon receiving $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$, Bob takes out the x-coordinate values K_{A1} and K_{A2} from T_{A1} and T_{A2}. Then, Bob verifies the equation:

$$e(S_A, P) \stackrel{?}{=} e(K_{A1} T_{A1} + K_{A2} T_{A2}, T_{A1}) e(T_{A2}, Y_A). \quad (1)$$

If this verification holds, Bob computes four shared session keys as follows:

$$K_1 = e(b_1 T_{A1}, Y_A + Y_B),$$
$$K_2 = e(b_2 T_{A1}, Y_A + Y_B),$$
$$K_3 = e(b_1 T_{A2}, Y_A + Y_B),$$
$$K_4 = e(b_2 T_{A2}, Y_A + Y_B).$$

Now, Alice and Bob have completed the protocol and got four shared session keys. For the correctness and security analysis the protocol refer to [4].

3. Weakness of Lee et al.'s protocol

3.1. Impersonation attack

We can see that, in the message sent by Alice to Bob $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$, S_A value is computed by using Alice's long-term private key. It implies that only Alice can produce S_A. However, we can analyze S_A as follows:

$$\begin{aligned} S_A &= (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + X_A T_{A2} = (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + X_A (a_2 P) = (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + a_2 (X_A P) \\ &= (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + a_2 Y_A. \end{aligned} \quad (2)$$

Checking the final equation in Eq. (2), we easily see that any attacker who wants to impersonate Alice could compute S_A directly from Alice's long-term public key without knowing Alice's long-term private key. The attacker does as follows:

- Choose two random integers a'_1 and a'_2 from Z_q^*.
- Compute $T'_{A1} = a'_1 P$ and $T'_{A2} = a'_2 P$.
- Take K'_{A1} and K'_{A2} as x-coordinate values of T'_{A1} and T'_{A2}, respectively.
- Compute $S'_A = (a'_1 K'_{A1} + a'_2 K'_{A2}) T_{A1} + a'_2 Y_A$.

The attacker could use the message $\{T'_{A1}, T'_{A2}, S'_A, Cert(Y_A)\}$ to impersonate Alice in order to share session keys with Bob. Because the value of S'_A is identical to the one computed by Alice as explained before, i.e., $S'_A = (a'_1 K'_{A1} + a'_2 K'_{A2}) T_{A1} + X_A T'_{A2}$, this value must pass the authentication check by Bob in **Step 4**, Eq. (1). Finally, the attacker could share four session keys with Bob under Alice's identity. Note that, the attacker could apply this impersonation attack to any entity of his/her choice. Since the roles of Alice and Bob are equivalent in the protocol, attacks on Alice also could be applied to Bob.

3.2. Vulnerability in perfect forward secrecy

Lee et al. [4] have claimed that the pairing-based authenticated key exchange protocol provides perfect forward secrecy, which means that any compromise of long-term private keys of Alice and Bob will not harm the previous session keys. However, this is not true in this case. When attackers know long-term private keys of Alice and Bob, X_A and X_B, respectively, the attackers easily compute the previous session keys as follows:

$$K_1 = e(a_1 T_{B1}, Y_A + Y_B) = e(T_{B1}, a_1(X_A + X_B)P) = e(T_{B1}, (X_A + X_B) T_{A1}).$$

Obviously, with the knowledge of X_A and X_B plus T_{A1} and T_{B1} are available in public, the attacker could compute K_1. By the same way, the attacker could get all remaining shared session keys.

4. Our revised version of the protocol

Based on our observation we have just made about why the attacks are feasible, we propose that our revised protocol should be modified in a minimal way. The setup phase is kept unchanged. We now describe the revised protocol.

Step S1. Alice $\rightarrow$ Bob: $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$.
Alice selects two random integers, $a_1, a_2 \in Z_q^*$, and computes $T_{A1} = a_1 P$ and $T_{A2} = a_2 P$. Let K_{A1} and K_{A2} be the x-coordinate values of T_{A1} and T_{A2}, respectively. Then, Alice computes S_A as follows:

$$S_A = (a_1 K_{A1} + a_2 K_{A2}) T_{A1} + X_A Y_B.$$

Alice sends the message $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$ to Bob. We assume that Bob's long-term public key, Y_B, can be easily obtained by Alice through a public directory.

Step S2. Bob $\rightarrow$ Alice: $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$.
Similarly, Bob randomly selects two integers, b_1 and b_2, and computes $T_{B1} = b_1 P$ and $T_{B2} = b_2 P$, where $b_1, b_2 \in Z_q^*$. Let K_{B1} and K_{B2} be the x-coordinate values of T_{B1} and T_{B2}, respectively. Then, Bob computes S_B as follows:

$$S_B = (b_1 K_{B1} + b_2 K_{B2}) T_{B1} + X_B Y_A.$$

Bob sends the message $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$ to Alice, where $Cert(Y_B)$ is the certificate of Bob's long-term public key signed by a trusted party. Similarly, Bob can obtain Alice's long-term public key, Y_A.

Step S3. Alice computes the shared session keys K_1, K_2, K_3, and K_4. Upon receiving $\{T_{B1}, T_{B2}, S_B, Cert(Y_B)\}$, Alice takes out the x-coordinate values K_{B1} and K_{B2} from T_{B1} and T_{B2}, separately. Alice checks the equation:

$$e(S_B, P) \stackrel{?}{=} e(K_{B1} T_{B1} + K_{B2} T_{B2}, T_{B1}) e(Y_A, Y_B).$$

If this verification holds, Alice computes four shared session keys as follows:

$$K_1 = e(a_1 T_{B1}, X_A T_{B1} + a_1 Y_B) = e(P, P)^{a_1 b_1 (b_1 X_A + a_1 X_B)},$$

$$K_2 = e(a_1 T_{B2}, X_A T_{B2} + a_1 Y_B) = e(P, P)^{a_1 b_2 (b_2 X_A + a_1 X_B)},$$

$$K_3 = e(a_2 T_{B1}, X_A T_{B1} + a_2 Y_B) = e(P, P)^{a_2 b_1 (b_1 X_A + a_2 X_B)},$$

$$K_4 = e(a_2 T_{B2}, X_A T_{B2} + a_2 Y_B) = e(P, P)^{a_2 b_2 (b_2 X_A + a_2 X_B)}.$$

Step S4. Bob computes the shared session keys K_1, K_2, K_3, and K_4.
Upon receiving $\{T_{A1}, T_{A2}, S_A, Cert(Y_A)\}$, Bob takes out the x-coordinate values K_{A1} and K_{A2} from T_{A1} and T_{A2}. Then, Bob verifies:

$$e(S_A, P) \stackrel{?}{=} e(K_{A1} T_{A1} + K_{A2} T_{A2}, T_{A1}) e(Y_B, Y_A).$$

Table 1
Performance evaluation.

Step	Lee et al. [4]	Our protocol
Computation of 2 short-term public keys	2S	2S
Computation of S_A or S_B	2S + 2M + A	2S + 2M + A
Verification	3e + 2S + A + M	3e + 2S + A + M
Key computation (1 key)	e + S + A	e + 3S + A
Available shared session keys	4	4

If this holds, Bob computes four shared session keys as follows:

$$\begin{aligned} K_1 &= e(b_1 T_{A1}, b_1 Y_A + X_B T_{A1}) = e(P,P)^{a_1 b_1 (b_1 X_A + a_1 X_B)}, \\ K_2 &= e(b_2 T_{A1}, b_2 Y_A + X_B T_{A1}) = e(P,P)^{a_1 b_2 (b_2 X_A + a_1 X_B)}, \\ K_3 &= e(b_1 T_{A2}, b_1 Y_A + X_B T_{A2}) = e(P,P)^{a_2 b_1 (b_1 X_A + a_2 X_B)}, \\ K_4 &= e(b_2 T_{A2}, b_2 Y_A + X_B T_{A2}) = e(P,P)^{a_2 b_2 (b_2 X_A + a_2 X_B)}. \end{aligned} \tag{3}$$

5. Analysis

Correctness. The correctness of shared keys is easily to notice by comparing key computation in **Steps S3** and **S4** in Section 4. The following is the correctness of S_A (similar for S_B) verification:

$$\begin{aligned} e(S_A, P) &= e((a_1 K_{A1} + a_2 K_{A2}) T_{A1} + X_A Y_B, P) = e((a_1 K_{A1} + a_2 K_{A2}) T_{A1}, P) e(Y_B, Y_A) = e((a_1 K_{A1} + a_2 K_{A2}) P, T_{A1}) e(Y_B, Y_A) \\ &= e(K_{A1} T_{A1} + K_{A2} T_{A2}, T_{A1}) e(Y_B, Y_A). \end{aligned}$$

Impersonation attack. Impersonation attack is infeasible since if an attacker wants to produce a forged message of Alice, the attacker has to compute S_A in order to pass Bob's verification. Given T_{A1}, T_{A2} of the attacker's choice, he/she still needs to compute $X_A Y_B = X_A X_B P$. However, computing $X_A X_B P$ from Y_A and Y_B is to solve the computational Diffie–Hellman problem in group G_1, which is believed to be computationally infeasible.

Known key security. Because random numbers are used in each round differently, the shared keys also differ for each round. Even the shared keys in a protocol session are exposed, attackers fail to relate these keys with the keys in other session since they are independent.

Key-compromise impersonation. If Alice's long-term private key is exposed, it does not enable an attacker to impersonate Bob to Alice. This can be eliminated since Alice uses Bob's public key in her shared secret keys computation. Even the attacker could masquerade the message sent to Alice in **Step S2** but ultimately, the attacker is unable to compute the shared keys without knowing Bob's long-term private key.

Perfect forward secrecy. In our protocol, when long-term private keys of both Alice and Bob, X_A and X_B are revealed, deriving session keys is still infeasible. Intuitively, we could see that, an attacker is given P, $T_{A1} = a_1 P$, $T_{B1} = b_1 P$ for instance, the attacker has to find out $e(P,P)^{a_1^2 b_1}$ and $e(P,P)^{a_1 b_1^2}$ from Eq. (3) in order to compute the shared key K_1. However, this is a Bilinear Square Diffie–Hellman problem [5] which is computationally infeasible.

Performance. The performance comparison between Lee et al.'s protocol and ours is presented in Table 1. In this table, S and A represent for scalar multiplication and point addition on an elliptic curve, respectively; e is pairing computation and M is the modular multiplication. As shown in this table, our revised protocol has the same computation compared with Lee et al.'s protocol at all steps except for the key computation. At this step, we require two more elliptic curve point multiplication operations in each key computation. However, this computation is negligible comparing with pairing computation. Therefore, we could consider the performance of the revised protocol and the original one is similar.

6. Conclusion

In this paper, we showed that Lee et al.'s authenticated multiple key exchange protocol based on bilinear pairings [4] fails to provide authenticity and perfect forward secrecy, despite of their claims to the contrary. We also provided a revised version of this protocol which prevents the attacks, but yet which does not add significantly to the communications or computational overhead for the protocol. Note that, bilinear pairings can provide beneficial properties, one has to carefully utilize them when designing cryptographic protocols.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments. This work was supported by the IT R&D program of MKE/TTA, 2009-P1-14-08I91, Mobile and next generation RFID technology standards development.

References

[1] Harn L, Lin H-Y. Authenticated key agreement without using one-way hash function. Electron Lett 2001;37(10):62930.
[2] Hwang Ren-Junn, Shiau Sheng-Hua, Lai Chih-Hua. An enhanced authentication key exchange protocol. Advanced information networking and applications, 2003. In: Proceedings of the 17th international conference on AINA 2003; 2729 March 2003. p. 2025.
[3] Lee Narn-Yih, Wu Chien-Nan. Improved authentication key exchange protocol without using one-way hash function. ACM Operat Syst Rev 2004;38(2):8592.
[4] Lee Narn-Yih, Wu Chien-Nan, Wang Chien-Chih. Authenticated multiple key exchange protocols based on elliptic curves and bilinear pairings. Comput Electr Eng 2008;34(1):12–20.
[5] Zhang F, Safavi-Naini R, Susilo W. An efficient signature scheme from bilinear pairings and its applications. In: Proceedings of international workshop on practice and theory in public key cryptography – PKC 2004. LNCS, vol. 2947. Springer-Verlag; 2004. p. 277–90.