Cryptanalysis of an efficient certificateless two-party authenticated key agreement protocol

Qingfeng Cheng

Luoyang University of Foreign Languages, Luoyang, PR China
qingfengc2008@sina.com

Abstract. Recently, He et al. (Computers and Mathematics with Applications, 2012, 64(6): 1914-1926) proposed a new efficient certificateless two-party authenticated key agreement protocol. They claimed their protocol was provably secure in the extended Canetti-Krawczyk (eCK) model. In this paper, we will show that their protocol is insecure. A type I adversary, who obtains one party's ephemeral private key, can impersonate the party to cheat the other party and compute the shared session key successfully. For overcoming this weakness, we also propose a simple countermeasure.

Key words: Authentication, Certificateless cryptography, Key agreement, Two-party, Ephemeral key compromise attack, Key replacement attack

1 Introduction

Al-Riyami and Paterson [1] first introduced the concept of certificateless public key cryptography (CL-PKC). In CL-PKC, each party can generate a long-term private key via combining its own secret key with a partial private key, which is issued by the key generation center (KGC). In general, two additional security attributes [2] need to be considered for CL-PKC:

- **Type I-key replacement attack resilience**: The malicious party, who knows nothing about a party's partial private key, cannot impersonate the party even if the party's secret key is compromised and the party's public key is replaced.
- **Type II-malicious KGC attack resilience**: The malicious KGC, who knows nothing about the party's private key and cannot repalce the party's public key, wants to impersonate the party.

Due to the nature of CL-PKC, it is not easy to design a secure certificateless authenticated key agreement (CL-AKA) protocol with good performance. The first CL-AKA protocol with a proof of security was presented by Lippold et al. [3] in 2009. However, their protocol is computationally too high. In 2011, Yang and Tan [4] proposed the first proven secure CL-AKA protocol without pairing,

which can avoid expensive costs. In 2012, He et al. [5] also proposed a pairing-free CL-AKA protocol. Unforunately, He et al.'s protocol is vulnerable to the type I adversary [6].

More recently, He et al. [7] proposed a new pairing-free CL-AKA protocol, called HPC protocol. They claimed their HPC protocol was provably secure in the eCK model [8] under the gap Diffie-Hellman assumption. In this paper, we will show that the HPC protocol is not secure. If a type I adversary can obtain the ephemeral private key, he can mount key replacement attack successfully. It means that the type I adversary with a party's ephemeral private key can impersonate the other party to cheat the party, who leaks the ephemeral private key. For overcoming this weakness, we also propose a simple countermeasure.

The rest of this paper is organized as follows. In Section 2, we briefly review the HPC protocol. In Section 3, we describe a combination attack on the HPC protocol. In Section 4, we propose a simple countermeasure against the combination attack. Finally, the conclusions will be given in Section 5.

2 Review of HPC protocol

In this section, we briefly review the HPC protocol proposed by He et al. in 2012. For more details, refer to [7].

2.1 System initialization stage

Let k be the security parameter, q be a k-bit prime, G be a cyclic group of order of q. P is a generator of G. Two secure hash functions $H_1 : \{0,1\}^* \times G \rightarrow Z_q^*$ and $H_2 : \{0,1\}^* \times \{0,1\}^* \times G^5 \rightarrow Z_q^*$. Key generation center (KGC) randomly chooses $x \in Z_q^*$ as the master private key and computes $P_{pub} = xP$ as the master public key. Finally, the KGC publishes $\{k, G, P, P_{pub}, H_1, H_2\}$ as system parameters and keeps the master private key s secretly.

2.2 Key extract stage

This phase is run by the KGC for each user with an identity $ID_i \in \{0,1\}^*$. The KGC first chooses $r_i \in Z_q^*$ randomly and computes $h_i = H_1(ID_i, R_i)$, where $R_i = r_iP$. Then the KGC computes $s_i = r_i + h_ix$. Finally, the KGC sends $\{s_i, R_i\}$ to each user with the identity ID_i via a secure channel.

The user with the identity ID_i chooses $x_i \in Z_q^*$ randomly, computes $P_i = x_i \cdot P$ and sets x_i as the secret value. Then the user with the identity ID_i sets $sk_i = (x_i, s_i)$ as the private key and $pk_i = P_i$ as the public key.

2.3 Key agreement stage

In the following description we suppose that two communications parties, A and B wish to communicate with each other.

1. Party A chooses $t_A \in Z_q^*$ randomly and computes $T_A = t_A P$. Then party A sends the message $M_A = (ID_A, R_A, T_A)$ to party B.
2. Upon receiving the message M_A, party B chooses $t_B \in Z_q^*$ randomly and computes $T_B = t_B P$. Then party B $M_B = (ID_B, R_B, T_B)$ to party A. Finally, party B computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T_B||K_{BA}^1||K_{BA}^2||K_{BA}^3),$$

where

$K_{BA}^1 = (t_B + s_B)(T_A + R_A + H_1(ID_A, R_A)P_{pub})$,
$K_{BA}^2 = (t_B + x_B)(T_A + P_A)$,
$K_{BA}^3 = t_B \cdot T_A$.

3. Upon receiving the message M_B, party A computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T_B||K_{AB}^1||K_{AB}^2||K_{AB}^3),$$

where
$K_{AB}^1 = (t_A + s_A)(T_B + R_B + H_1(ID_B, R_B)P_{pub})$,
$K_{AB}^2 = (t_A + x_A)(T_B + P_B)$,
$K_{AB}^3 = t_A \cdot T_B$.

3 Attack on HPC protocol

In this section, we will show that the HPC protocol is not secure in the eCK model. It also means that the HPC protocol fails to provide implicit authentication.

The type I adversary E first replaces party B's pulic key as $P'_B = H_1(ID_B, R'_B)P_{pub} + P$. If the type I adversary E can obtain party A's ephemeral private key t_A, then he can mount the attack on the HPC protocol as follows:

1. Party A chooses $t_A \in Z_q^*$ randomly and computes $T_A = t_A P$. Then party A sends the message $M_A = (ID_A, R_A, T_A)$ to party B.
2. Upon intercepting the message M_A, the adversary E chooses $t_E \in Z_q^*$ randomly and computes $T'_B = -H_1(ID_B, R'_B)P_{pub}$, where $R'_B = t_E P$. Then the adversary E impersonates party B and sends $M'_B = (ID_B, R'_B, T'_B)$ to party A. Finally, the adversary E computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T'_B||K_{EA}^1||K_{EA}^2||K_{EA}^3),$$

where

$K_{EA}^1 = r_E(t_A P + s_A P)$,
$K_{EA}^2 = t_A P + x_A P$,
$K_{EA}^3 = t_A \cdot T'_B$.

3. Upon receiving the message M_B, party A computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T'_B||K^1_{AB}||K^2_{AB}||K^3_{AB}),$$

where

$$K^1_{AB} = (t_A + s_A)(T'_B + R'_B + H_1(ID_B, R'_B)P_{pub}),$$
$$K^2_{AB} = (t_A + x_A)(T'_B + P'_B),$$
$$K^3_{AB} = t_A \cdot T'_B.$$

Since the adversary E has obtained t_A and $x_A P$ is public, he can compute K^2_{EA} and K^3_{EA}. In addition, the adversary E also computes K^1_{EA} with the random number r_E, which is chosen by the adversary.

Further, we have

$$\begin{aligned}
K^1_{AB} &= (t_A + s_A)(T'_B + R'_B + H_1(ID_B, R'_B)P_{pub}) \\
&= (t_A + s_A)(-H_1(ID_B, R'_B)P_{pub} + t_E P + H_1(ID_B, R'_B)P_{pub}) \\
&= (t_A + s_A)t_E P \\
&= K^1_{EA}, \\
K^2_{AB} &= (t_A + x_A)(T'_B + P'_B) \\
&= (t_A + x_A)(-H_1(ID_B, R'_B)P_{pub} + H_1(ID_B, R'_B)P_{pub} + P) \\
&= (t_A + x_A)P \\
&= K^2_{EA}, \\
K^3_{AB} &= t_A \cdot T'_B = K^3_{EA}.
\end{aligned}$$

It means that the adversary E and party A will generate the same session key. Now, the adversary E has successfully cheated party A to believe that he has shared secret session key with party B. However, party B does not involve in this session completely. It means that the HPC protocol is insecure against the combination of the public key replace attack and ephemeral key compromise attack.

4 Countermeasure

In this section, we propose a simple countermeasure. The improved HPC protocol will be able to resist the combination attack and also keep the original HPC protocol's security attributes. Since the system initialization stage and the key extract stage are the same as the original HPC protocol, we only present the key agreement stage.

The improved key agreement stage is described as follows:

1. Party A chooses $t_A \in Z_q^*$ randomly and computes $T_A = t_A P$. Then party A sends the message $M_A = (ID_A, R_A, T_A)$ to party B.
2. Upon receiving the message M_A, party B chooses $t_B \in Z_q^*$ randomly and computes $T_B = t_B P$. Then party B $M_B = (ID_B, R_B, T_B)$ to party A. Finally, party B computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T_B||K_{BA}^1||K_{BA}^2||K_{BA}^3),$$

where

$K_{BA}^1 = (t_B + s_B)(T_A + R_A + H_1(ID_A, R_A)P_{pub})$,
$K_{BA}^2 = (t_B + x_B H_1(ID_B, ID_A, T_B, P_B))(T_A + H_1(ID_A, ID_B, T_A, P_A)P_A)$,
$K_{BA}^3 = t_B \cdot T_A$.

3. Upon receiving the message M_B, party A computes the shared session key as follows:

$$sk = H_2(ID_A||ID_B||T_A||T_B||K_{AB}^1||K_{AB}^2||K_{AB}^3),$$

where

$K_{AB}^1 = (t_A + s_A)(T_B + R_B + H_1(ID_B, R_B)P_{pub})$,
$K_{AB}^2 = (t_A + x_A H_1(ID_A, ID_B, T_A, P_A))(T_B + H_1(ID_B, ID_A, T_B, P_B)P_B)$,
$K_{AB}^3 = t_A \cdot T_B$.

Attack resilience

The improved HPC protocol can resist the combination attack. Even if the adversary E, with party A's ephemeral private key, replaces party B's pulic key as $P'_B = H_1(ID_B, R'_B)P_{pub} + P$, chooses $t_E \in Z_q^*$ randomly and computes $T'_B = -H_1(ID_B, R'_B)P_{pub}$, where $R'_B = t_E P$, he only can compute K_{AB}^1 and K_{AB}^3. However, the adversary E cannot compute the shared session key, since he cannot compute K_{AB}^2.

5 Conclusions

In this paper, we analyze the HPC protocol, which is proposed by He et al. in 2012. Our work shows that the HPC protocol is vulnerable to key replacement attack and ephemeral key compromise attack. The adversary can successfully cheat a honest participant of the HPC protocol to believe that he has generated the shared session key with another honest participant, who may be completely uninvolved in the session. For preventing the combination attack, we also propose a slight modification to the original HPC protocol.

References

1. S. Al-Riyami, K.G. Paterson, Certificateless public key cryptography, in: Proceedings of ASIACRYPT 2003, in: LNCS, vol. 2894, Springer-Verlag, 2003, pp. 452-473.
2. M. Au, J. Chen, J. Liu, Y. Mu, D. Wong, G. Yang, Malicious KGC attacks in certificateless cryptography, in: Proceedings of the 2007 ACM Symposium on Information, Computer and Communications Security, 2007, pp. 302-311.
3. G. Lippold, C. Boyd, J. Nieto, Strongly secure certificateless key agreement, in: Proceedings of Pairing 2009, in: LNCS, vol. 5671, Springer-Verlag, 2009, pp. 206-230.
4. G. Yang, C. Tan, Strongly secure certificateless key exchange without pairing, in: 6th ACM Symposium on Information, Computer and Communications Security, 2011, pp. 71-79.
5. D. He, J. Chen, J. Hu, A pairing-free certificateless authenticated key agreement protocol, International Journal of Communication Systems 25(2)(2012) 221-230.
6. D. He, Y. Chen, J. Chen, R. Zhang, W. Han, A new two-round certificateless authenticated key agreement protocol without bilinear pairings, Mathematical and Computer Modelling 54(11-12)(2011)3143-3152.
7. D. He, S. Padhye, J. Chen. An efficient certificateless two-party authenticated key agreement protocol. Computers and Mathematics with Applications 64(6)(2012)1914-1926.
8. B. LaMacchia, K. Lauter, A. Mityagin, Stronger security of authenticated key exchange, in: Proceedings of ProvSec 2007, in: LNCS, vol. 4784, Springer-Verlag, 2007, pp. 1-16.