Certificateless One-Way Authenticated Two-Party Key Agreement Protocol

Wuping Chen*, Lei Zhang†, Bo Qin†, Qianhong Wu*‡ and Huanguo Zhang*
*School of Computer, Wuhan University, Wuhan City, China
Email: liss@whu.edu.cn
†Dept. of Comp. Eng. and Maths, Uni. Rovira i Virgili, Tarragona, Spain

Abstract

Key agreement is one of the fundamental cryptographic primitives in public key cryptography. It plays an important role for securing systems in practice. In this paper, we present the first certificateless One-Way authenticated Two-Party key agreement protocol. The security of the proposed protocol is analyzed based on the intractability of the standard discrete logarithm (DL) and bilinear Diffie-Hellman (BDH) problems. For efficiency, our protocol enjoys low complexity in both communication and computation.

1. Introduction

In two-party private communication environment, a session key is usually required to achieve the goal of encrypting a message by a sender and decrypting the message by a receiver. In practice, before two parties begin private communication, they should firstly establish a session key between them. There are two different approaches to establish a session key between entities. One is known as enveloping or key transport. The other is known as key agreement. In the enveloping or key transport approach, the session key is generated by the sender itself and then transported to the receiver. While in the key agreement approach, both entities may contribute information from which a joint secret key is derived as the session key.

The first practical solution to key agreement problem is the Diffie-Hellman key exchange protocol [6]. However the Diffie-Hellman protocol does not provide authentication to the two communication entities, and hence subjects to man in the middle attack [9]. Over the years, there have been many attempts to add authentication to the Diffie-Hellman protocol [10], [13] as well as to develop new key agreement protocols [17]. The research in this area has been focusing on the design of authenticated key agreement (AK for short) protocols with lower computation and communication cost and round complexity.

As described in [11], there are three types of two-party authenticated key agreement protocols, namely, non-interactive, one-way and one-round or two pass. If both entities require to transmit information to each other during the protocol, then the protocol is called one-round or two pass. If only one entity is required to transmit information to the other during the protocol, then the protocol is called one-way. If no information need to be transmitted between two entities, the protocol is called non-interactive. In most situations, it is very desirable to build key agreement protocols with high security but a minimal number of passes (and rounds) of communication and low computation cost. And of all kinds of key agreement protocols, the one-way key agreement protocols are the most suitable protocols for network environment where the communication cost is the critical consideration.

Key agreement protocols may be designed under different public key cryptosystems. There are mainly three types of public key cryptosystems, namely conventional Certificate-based, ID-based, and certificateless public key cryptosystems. A number of key agreement protocols have been proposed under the conventional discrete logarithm based public key systems. However, the management of public key certificates requires a large amount of computation, storage, and communication cost in this kind of systems.

In 1984, Shamir [12] introduced the Identity-based Public Key Cryptography (ID-PKC for short) to eliminate the requirement of certificates. Subsequently, a number of proposals were presented to instantiate the notion of identity based key agreement protocols [3], [4], [5], [7]. In such proposals, the PKG is unconditionally trusted and the ID-PKC protocols suffer from a key escrow problem. This may be undesirable in some scenarios where it is difficult to find such a party fully trusted by the distributed users.

Certificateless public key cryptography is a new paradigm which was first introduced by Al-Riyami and Paterson [1] in 2003. Their main purpose is to solve the key escrow problem in ID-PKC [12], while keeping the implicit certification property of ID-PKC. Like ID-PKC, certificateless public key cryptography does not use any public key certificate. Up to now, some secure and efficient certificateless encryption or signature schemes have been proposed [1], [2], [8], [14]. However, only a little attention has been paid to key agreement protocols in certificateless public key settings.

The first certificateless two-party key agreement protocol was proposed by Al-Riyami and Paterson [1]. Their protocol requires both entities to transmit information to the other and hence is one round. Later several certificateless two-party key agreement protocols have been presented [15], [16]. These protocols are also one-round and as far as we know, no
2. Preliminaries

2.1. Bilinear Pairings

Let $G_1$ be an additive group of prime order $q$ and $G_2$ be a multiplicative group of the same order. Let $P$ denote a generator of $G_1$. A mapping $e : G_1 \times G_1 \rightarrow G_2$ is called a bilinear mapping if it satisfies the following properties:

1. Bilinear: $e(aP, bQ) = e(P, Q)^{ab}$ for all $P, Q \in G_1, a, b \in Z_q^*$.
2. Non-degeneracy: There exists $P, Q \in G_1$ such that $e(P, Q) \neq 1$.
3. Computable: There exists an efficient algorithm to compute $e(P, Q)$ for any $P, Q \in G_1$.

2.2. Mathematical Problems

Here we present some mathematical problems, which form the basis of security for our key agreement protocols.

**Definition 1. Discrete Logarithm (DL) Problem:** Let $G = (< g >)$ be a cyclic group of order $q$ generated by $g$. The Discrete Logarithm (DL) Problem in $G$ is: Given an arbitrary element $\alpha \in G$, to find an integer $a \in Z_q^*$ such that $\alpha = g^a$.

Let $G_1, G_2$, and $e : G_1 \times G_1 \rightarrow G_2$ be groups and bilinear mapping as specified in Section 2.1.

**Definition 2. Bilinear Diffie-Hellman (BDH) Problem:**
Given a randomly chosen $P \in G_1$, as well as $aP, bP, cP$ (for random unknown $a, b, c \in Z_q^*$), to compute $e(P, P)^{abc}$.

**Definition 3. Decisional Bilinear Diffie-Hellman (DBDH) Problem:**
Given a randomly chosen $P \in G_1$, as well as $aP, bP, cP$ (for random unknown $a, b, c \in Z_q^*$) and $h \in G_2$, decide whether $h = e(P, P)^{abc}$.

**Definition 4. Gap Bilinear Diffie-Hellman (GBDH) Problem:**
Given a randomly chosen $P \in G_1$, as well as $aP, bP$ and $cP$ (for random unknown $a, b, c \in Z_q^*$), compute $e(P, P)^{abc}$ with the help of the DBDH oracle.

2.3. Certificateless Key Agreement Protocol

A certificateless key agreement protocol is defined by six algorithms: **Setup**, **Partial-Private-Key-Extract**, **Set-Secret-Value**, **Set-Private-Key**, **Set-Public-Key**, and **Key-Agreement**. The description of each algorithm is as follows.

- **Setup:** An algorithm runs by the KGC that accepts a security parameter $k$ and returns a master-key and a list of system parameters $\text{params}$.
- **Partial-Private-Key-Extract:** An algorithm runs by the KGC that accepts a user’s identity $ID_i$, a parameter list $\text{params}$ and a master-key to produce the user’s partial private key $D_i$.
- **Set-Secret-Value:** An algorithm runs by a user that accepts a parameter list $\text{params}$ and a user’s identity $ID_i$ to produce the user’s secret value $x_i$.  
- **Set-Private-Key:** An algorithm runs by a user that takes as input a parameter list $\text{params}$, the user’s identity $ID_i$, partial private key $D_i$ and the user’s secret value $x_i$ to produce a private key $P_i$ for the user.
- **Set-Public-Key:** An algorithm runs by a user that takes as input a parameter list $\text{params}$, a user’s identity $ID_i$ and the user’s secret value $x_i$ to produce a public key $K_i$ for the user.
- **Key-Agreement:** This algorithm takes as input a parameter list $\text{params}$, $(S_A, ID_A, P_A)$ for sender $A$, $(S_B, ID_B, P_B)$ for receiver $B$ to produce the session key $K$, $K_{AB}$ for $A$ and $K_{BA}$ for $B$, where $K = K_{AB} = K_{BA}$.

3. Desirable Attributes

It is desirable for AK protocols to possess the following security attributes as described in [11].

1. **Known-key security**: Each run of the protocol should result in a unique secret session key. The compromise of one session key should not compromise other session keys.
2. **Unknown key-share**: An entity $A$ must not be coerced into sharing a key with any entity $C$ if $A$ thinks that she is sharing the key with another entity $B$.
3. **No key control**: Neither entity should be able to force the session key to be a preselected value.
4. **Sender’s key-compromise impersonation**: The compromise of sender $A$’s private key will allow an adversary to impersonate $A$, but it should not enable the adversary to impersonate other entities in the presence of $A$.
5. **Sender’s forward security**: If private keys of senders are compromised, the secrecy of previously established session keys should not be affected.
6. **Random number compromise security**: The compromise of a random number or a random element se-
lected by the sender \(A\) should not compromise \(A\)'s private key or the established session keys.

4. Our Protocols

In this section, we present a certificateless authenticated key agreement protocol. The constructions are as follows:

- **Setup:** Let \(G_1\) be a cyclic additive group generated by \(P\), whose order is a prime \(q\), \(G_2\) be a cyclic multiplicative group of the same order \(q\), and \(e : G_1 \times G_1 \rightarrow G_2\) be a bilinear pairing. This algorithm runs as follows:
  1. Choose a random master-key \(s \in Z_q^*\) and set \(P_0 = sP\).
  2. Choose a cryptographic hash function \(H : \{0,1\}^* \rightarrow G_1, H_1 : G_1 \rightarrow Z_q^*, H_2 : \{0,1\}^* \rightarrow \{0,1\}^n\).

The system parameters \(\text{params} = (G_1, G_2, e, P, P_0, H, H_1, H_2)\).

- **Partial-Private-Key-Extract:** This algorithm accepts an identity \(ID_i \in \{0,1\}^n\) and selects a random \(x_i \in Z_q^*\). It outputs \(x_i\) as the user’s secret value.

- **Set-Secret-Value:** This algorithm takes as input \(\text{params}\) and a user’s identity \(ID_i\), and selects a random \(x_i \in Z_q^*\). The output of the algorithm is the private key \(S_i = (x_i, D_i)\).

- **Set-Private-Key:** This algorithm takes as input \(\text{params}\), a user’s partial private key \(D_i\), and the user’s secret value \(x_i \in Z_q^*\). The output of the algorithm is the private key \(S_i = (x_i, D_i)\).

- **Set-Public-Key:** This algorithm accepts \(\text{params}\) and a user’s secret value \(x_i \in Z_q^*\) to produce the user’s public key \(P_i = x_iP\).

- **Key-Agreement:** Assume the sender \(A\) has the private key \(S_A = (x_A, D_A)\) and public key \(P_A = x_AP\). The receiver \(B\) has the private key \(S_B = (x_B, D_B)\) and public key \(P_B = x_BP\). The protocol runs as follows:

\[
\begin{array}{c|c}
\text{Sender } A & \text{Receiver } B \\
\hline
r_A \in_R Z_q^* & U = rQ_A \\
U & K_{AB} \\
\hline
K_{BA} & \\
\end{array}
\]

First \(A\) picks a random \(r \in Z_q^*\), computes \(U = rQ_A\), and sends \(U\) to \(B\). Then \(A\) and \(B\) can establish their session key as follows:

- \(A\) first computes
  \[h = H_1(U), Q_B = H(ID_B),\]
  then computes
  \[K_{AB} = H_2(ID_A, ID_B, P_A, P_B, U, rP_B, \alpha),\]
where \(\alpha = (e(D_A, Q_B)e(Q_A, P_B)x_A)^{r+h}\).
- \(B\) first computes:
  \[h = H_1(U), Q_A = H(ID_A),\]
then computes
  \[K_{BA} = H_2(ID_A, ID_B, P_A, P_B, U, x_BU, \beta),\]
where \(\beta = e(U + hQ_A, x_BP_A + D_B)\).

**Consistency:** It is easy to see \(K_{AB} = K_{BA}\) holds, since
\[
\begin{align*}
\alpha &= (e(D_A, Q_B)e(Q_A, P_B)x_A)^{r+h} \\
&= (e(D_A, Q_B)e(Q_A, P_B)x_A)^r(e(D_A, Q_B)e(Q_A, P_B)x_A)^h \\
&= e(Q_A, Q_B)^{sr}e(Q_A, P)^{rx_Ax_B}e(hQ_A, Q_B)^{se}(hQ_A, P)^{x_Ax_B} \\
&= e(rQ_A, sQ_B)e(rQ_A, x_AP)^{x_B}e(hQ_A, sQ_B)e(hQ_A, x_Bx_AP) \\
&= e(U, D_B)e(U, P_A)^{x_B}e(hQ_A, D_B)e(hQ_A, x_BP_A) \\
&= e(U, x_BP_A + D_B)e(hQ_A, x_BP_A + D_B) \\
&= e(U + hQ_A, x_BP_A + D_B) \\
&= e(U + hQ_A, x_BP_A + D_B) = \beta.
\end{align*}
\]

5. Security Analysis

In this section, we show that our protocol possesses all the security attributes described in Section 3.

Two types of adversaries [1] with different capabilities are generally considered in CL-PKC. They are known as Type I Adversaries and Type II Adversaries. A Type I Adversary \(A_I\) has the ability to replace the public key of any user with a value of his choice, but he does not have access to KGC’s master-key. While a Type II Adversary \(A_{II}\) has access to the master-key (which is used to generate a user’s partial private key) but cannot perform public key replacement. In the following, we define \(E \in \{A_I, A_{II}\}\) Now we show that our protocol is secure against both \(A_I\) and \(A_{II}\).

1) **Known-key security:** This comes from the fact, each run of the protocol between two entities \(A\) and \(B\), a random \(r\) is selected. A session key as a result is distributed uniformly in \(\{0,1\}^n\) with no connection to other session keys.

2) **Unknown key-share to both types of adversaries:** \(A\) can’t be coerced into sharing a key with any entity \(E\) if \(A\) thinks that she is sharing the key with \(B\). This comes from the fact, \(A\) explicitly uses \(B\)'s identity \(ID_B\) and public key \(P_B\) in her contribution to the session key. \(B\) can’t be coerced into sharing a key with any entity \(E\) if \(B\) thinks that she is sharing the key with \(A\) either. This comes from the fact, \(B\) explicitly uses \(A\)'s identity \(ID_A\) and public key \(P_A\) in her contribution to the session key. Furthermore, the value \(U + hQ_A\) (where \(U = rQ_A\)) can not be predetermined for example by replacing \(rQ_A + hQ_A\) to \(r'Q_E\) for some known \(r'\).

3) **No key control:** Since the value \(r\) is selected by \(A\), it is easy to see that \(B\) can’t control the session key. \(A\) can’t do this either comes from the fact that for a predetermined session key \(K\) to
find $r$ such that $H_2(ID_A, ID_B, P_A, P_B, U, \alpha) = K$ is computationally impossible, where $\alpha = (e(D_A, Q_B)e(Q_A, P_B)^{x_A})^{r+h}$.

4) Sender's key-compromise impersonation: This comes from the fact that without knowing the value $r$, although the value $V = e(D_A, Q_B)e(Q_A, P_B)^{x_A}$ can be computed by an adversary $E$, the session key $K = H_2(ID_A, ID_B, P_A, P_B, U, V^{r+h})$ can't be computed by $E$. Hence, it is impossible for $E$ to impersonate any other entities like $B$ to $A$ even he knows $S_A$.

5) Sender's forward security to both types of adversaries: Although the value $V = e(D_A, Q_B)e(Q_A, P_B)^{x_A}$ can be computed by an adversary $E$, the forward session key $K = H_2(ID_A, ID_B, P_A, P_B, U, V^{r+h})$ with unknown $r$ can't be computed by $E$, from since from $U = rQ_A$ to compute $r$ the adversary is equivalent to solving the DL problem. Hence, our scheme has sender's forward security to both types of adversaries.

6) Random number compromise security to both types of adversaries: This comes from the fact that a type I adversary without the knowledge of $D_A$ and $D_B$ has no advantage to compute $e(Q_A, Q_B)^x$, since it is equal to solve the BDH problem and for a type II adversary, without the knowledge of $x_A$ and $x_B$, he has no advantage to compute $e(Q_A, P_B)^{x_A x_B}$, since it is equal to solve the BDH problem either.

Acknowledgment and Disclaimer

This paper is partly supported by the Chinese NSF project 60673071, the Chinese 863 projects 2006AA01Z442 and 2007AA01Z411, and by the Spanish projects CONSOLIDER INGENIO 2010 CSD2007-00004 “ARES” and TSI2007-65406-C03-01 “E-AEGIS”. The views of those authors with the UNESCO Chair in Data Privacy do not necessarily reflect the position of UNESCO nor commit that organization.

References


