Certificateless Authenticated Two-Party Key Agreement Protocols

Tarjei K. Mandt and Chik How Tan

Norwegian Information Security Laboratory
Department of Computer Science and Media Technology
Gjøvik University College
P.O. Box 191, N-2802 Gjøvik, Norway
{tarjei.mandt,chik.tan}@hig.no

Abstract. In their seminal paper on certificateless public key cryptography, Al-Riyami and Paterson (AP) proposed a certificateless authenticated key agreement protocol. Key agreement protocols are one of the fundamental primitives of cryptography, and allow users to establish session keys securely in the presence of an active adversary. AP's protocol essentially requires each party to compute four bilinear pairings. Such operations can be computationally expensive, and should therefore be used moderately in key agreement. In this paper, we propose a new certificateless authenticated two-party key agreement protocol that only requires each party to compute two pairings. We analyze the security of the protocol and show that it achieves the desired security attributes. Furthermore, we show that our protocol can be used to establish keys between users of different key generation centers.

Keywords: Certificateless public key cryptography, authenticated key agreement, bilinear map.

1 Introduction

In the public key infrastructure (PKI), certificates are used to provide an assurance of the relationship between the public keys and the identities that hold the corresponding private keys. However, there are many problems associated with certificates such as revocation, storage, distribution, and cost of validation. In 1984, Shamir [7] proposed the notion of identity-based public key cryptography (ID-PKC) to simplify certificate management. The idea of ID-PKC is to let an entity's public key be directly derived from certain aspects of its identity, such as the IP address of the hostname or the e-mail address. Thus, ID-PKC also eliminates the need for certificates.

Unfortunately, ID-PKC is not without problems. Identity-based systems rely on a private key generator (PKG) that uses a system-wide master key in generating private keys. Thus, many identity-based schemes inevitably introduce *key escrow*: the PKG can recover the session key established by entities for which it has issued a private key. This property is either acceptable or unacceptable. For instance, in the health care profession it may be a legal requirement to provide

M. Okada and I. Satoh (Eds.): ASIAN 2006, LNCS 4435, pp. 37–44, 2007.

an audit trail to every transaction. On the other hand, such invasion of privacy may cause ID-PKC to be unsuited in a variety of other applications, such as personal communications.

Certificateless public key cryptography (CL-PKC) [1] was proposed by Al-Riyami and Paterson to alleviate the problems associated with PKI and ID-PKC. It does not require the use of certificates and yet does not have the key escrow limitation of ID-PKC. For this reason, CL-PKC can be seen as a public key cryptography model intermediate between the two former paradigms.

In their seminal paper, Al-Riyami and Paterson (AP) proposed a certificateless authenticated two-party key agreement protocol. Key agreement protocols allow entities to establish session keys securely in the presence of an active adversary. AP's protocol essentially requires each party to compute four bilinear pairings. Such operations can be computationally expensive (for instance, on low-power devices) and should therefore be used moderately in key agreement. Moreover, AP's protocol also requires users to exchange public keys comprising two group elements. Ideally, public keys should only comprise one group element as in identity-based key agreement.

This paper proposes a new certificateless authenticated two-party key agreement protocol [5] that is more efficient than AP's protocol. Each entity involved in the protocol is only required to compute two pairings, and the public keys exchanged by the entities only comprise one group element. As public keys are not bound to a specific key generation center (KGC), the protocol can also be used to establish session keys between users of different KGCs. Furthermore, we show that the protocol achieves the security attributes that are desired in authenticated key agreement.

The rest of the paper is organized as follows. In Section 2 we give underlying definitions and define the security attributes of authenticated key agreement. In Section 3 we propose a new certificateless authenticated two-party key agreement protocol, and in Section 4 we show how the protocol can be used by entities of different KGCs. In Section 5 and 6 we analyze the security and the efficiency of the protocol respectively, and in Section 7 we provide a conclusion of the paper.

2 Preliminaries

2.1 Bilinear Pairings

Let $\mathbb{G}_1$ be an additive group with a large prime order q and let $\mathbb{G}_2$ be a multiplicative group of the same order. An *admissible pairing* e is then a function $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ that satisfies the following properties:

1. *Bilinearity*: For all $P, Q \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_q^*$, $e(aP, bQ) = e(P, Q)^{ab}$.
2. *Non-degeneracy*: There exists a $P \in \mathbb{G}_1$ such that $e(P, P) \neq 1$.
3. *Computability*: There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}_1$.

The admissible pairing e can be derived from a Weil or Tate pairing on an elliptic curve over a finite field. For further details, see [3].

2.2 Diffie-Hellman Problems

The security of the proposed protocols is based on a set of well-studied problems that are assumed to be hard to compute efficiently. These problems are defined as follows:

Definition 1 (Discrete Logarithm Problem). *Given $Q \in \mathbb{G}_1$ where P is a generator of $\mathbb{G}_1$, find an element $a \in \mathbb{Z}_q^*$ such that $aP = Q$.*

Definition 2 (Computational Diffie-Hellman Problem). *Let P be a generator of $\mathbb{G}_1$. Given $\langle P, aP, bP\rangle \in \mathbb{G}_1$ where $a, b \in \mathbb{Z}_q^*$, compute abP.*

Definition 3 (Bilinear Diffie-Hellman Problem). *Let e be a bilinear pairing on $(\mathbb{G}_1, \mathbb{G}_2)$ and P be a generator of $\mathbb{G}_1$. Given $\langle P, aP, bP, cP\rangle \in \mathbb{G}_1$ where $a, b, c \in \mathbb{Z}_q^*$, compute $e(P, P)^{abc} \in \mathbb{G}_2$.*

We assume that the order q of the groups $\mathbb{G}_1$ and $\mathbb{G}_2$ is large enough to make solving the discrete logarithm problem computationally infeasible.

2.3 Security of Authenticated Key Agreement

A key agreement protocol is said to be *authenticated* (AK) if it ensures authenticity of the involved parties. Specifically, a party can only compute a shared key if it holds the claimed identity. Thus, by mutually proving possession of the shared key, each party may be assured that the peer is a legitimate entity. The proposed protocols of this paper will use cryptographic *message authentication codes* in providing such assurance.

Definition 4 (Message Authentication Code). *A message authentication code $\mathtt{MAC} = (\mathcal{K}_{\mathtt{mac}}, \mathcal{T}_{\mathtt{mac}}, \mathcal{V}_{\mathtt{mac}})$ consists of three algorithms: key generation $\mathcal{K}_{\mathtt{mac}}$, message authentication $\mathcal{T}_{\mathtt{mac}}$ and verification $\mathcal{V}_{\mathtt{mac}}$. The key generation algorithm $\mathcal{K}_{\mathtt{mac}}$ generates a key $k \leftarrow \mathcal{K}_{\mathtt{mac}}$. The message authentication algorithm $\mathcal{T}_{\mathtt{mac}}$ returns an authentication tag $\tau = \mathcal{T}_{\mathtt{mac}}(k, m)$ for given a key k and a message m. The verification algorithm $\mathcal{V}_{\mathtt{mac}}$ returns 1 for "accept" and 0 for "reject", for given a key k, the message m and the authentication tag τ.*

Furthermore, it is desired that AK protocols possess a number of *security attributes* [2, 4]:

- **Known session key security**. Each run of the key agreement protocol should result in a unique secret session key. An adversary who learns a session key should not be able to recover data from past or future sessions.
- **Forward secrecy**. If long-term private keys of one or more entities are compromised, the secrecy of previous session keys established by these entities should not be affected. In the presence of a key generation center (KGC) with a system-wide master key, *KGC forward secrecy* implies that compromise of this key should not reveal previously established session keys.

- **Key-compromise impersonation**. If A's long-term private key is compromised, the adversary can impersonate A, but the adversary should not be able to impersonate other entities to A.
- **Unknown key-share**. Entity A should not be coerced into sharing a key with entity C when, in fact, A thinks she is sharing a key with entity B.
- **Key control**. Neither party involved in a protocol run should be able to control the outcome of the session key more than the other.
- **Known session-specific temporary information security**. Many protocols use some randomized private input to produce a unique session key in each run of a protocol. Exposure of such private temporary information should not compromise the secrecy of the generated session key.

3 Proposed Certificateless Authenticated Key Agreement

In this section, we will present a new certificateless authenticated two-party key agreement protocol based on pairings. As noted earlier, certificateless key agreement does not rely on certificates, nor does it employ a key generation center (KGC) that knows every user's private key.

Setup: Let H_1 and H_2 be two independent key derivation functions such that $H_1 : \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \{0,1\}^k$ and $H_2 : \mathbb{G}_2 \times \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \{0,1\}^l$ for some integers $k, l > 0$. Let also H be a Map-To-Point [3] function such that $H : \{0,1\}^* \rightarrow \mathbb{G}_1$. The KGC randomly selects its secret master key $s \in \mathbb{Z}_q^*$ and computes the public key $P_0 = sP$ where $P \in \mathbb{G}_1$ is a public generator. The KGC then publishes the system parameters $\langle \mathbb{G}_1, \mathbb{G}_2, e, q, P, P_0, H, H_1, H_2 \rangle$.

Private Key Extraction: For any given entity A with identity $\mathtt{ID}_A \in \{0,1\}^*$, the long-term private/public key pair is generated as follows:

1. A randomly selects a secret value $x_A \in \mathbb{Z}_q^*$.
2. The KGC generates A's *partial* private key $D_A = sQ_A$ where A's identifier $Q_A = H(\mathtt{ID}_A) \in \mathbb{G}_1$. A may check the validity of D_A by verifying that $e(Q_A, P_0) = e(D_A, P)$.
3. A's (full) private key is given by $S_A = \langle D_A, x_A \rangle$. A's public key is computed as $P_A = x_A P$.

For entity B, the private key extraction is similar to A.

Key Agreement: In order to jointly establish a session key, entities A and B randomly select the short-term session-specific private keys $a, b \in \mathbb{Z}_q^*$ and compute the corresponding short-term public keys $T_A = aP$ and $T_B = bP$ respectively. They then exchange the following messages:

$$(1)\ A \rightarrow B : T_A, P_A$$
$$(2)\ B \rightarrow A : T_B, P_B, \mathcal{T}_{\mathtt{mac}}(k', T_B, P_B)$$
$$(3)\ A \rightarrow B : \mathcal{T}_{\mathtt{mac}}(k', T_A, P_A)$$

Both entities validate each other's public key by testing the group membership $P_A, P_B \in \mathbb{G}_1^*$. The MAC algorithm $\mathcal{T}_{\mathtt{mac}}$ is used to generate an authentication tag τ of the shared key k' (defined below) and the transmitted

message. If a party fails to verify a received authentication tag, then the protocol run is terminated. A and B compute the session key as follows:

$$K_A = e(Q_B, P_0 + P_B)^a \cdot e(D_A + x_A Q_A, T_B)$$
$$K_B = e(D_B + x_B Q_B, T_A) \cdot e(Q_A, P_0 + P_A)^b$$

The scheme is consistent because:

$$K = K_A = K_B = e(Q_B, P)^{a(s+x_B)} \cdot e(Q_A, P)^{b(s+x_A)}$$

In order to ensure that an attacker cannot gain any information from the session key, A and B use a key derivation function H on K, abP, and $x_A x_B P$. Thus, the final session key is given by $k = H_1(K \| abP \| x_A x_B P)$. The MAC key $k' = H_2(K \| abP \| x_A x_B P)$ is different from k in order to provide key indistinguishability [2].

Note that both parties are authenticated through the verification of an authentication tag τ (provided by the peer) when they successfully establish a shared key. Thus, both parties are assured that the peer holds the claimed identity. If no such assurance had been provided, an adversary could possibly engage in a key agreement with B, while impersonating A. This would lead B to falsely believe a session key is established with A.

Furthermore, note that the structure of the long-term private/public key pair differs from Al-Riyami and Paterson's protocol [1]. Specifically, the public key P_A only comprises one element of $\mathbb{G}_1$ and no longer binds an entity to a specific KGC, thus allowing protocol participants under different trusted authorities to establish keys. Also see that S_A separates D_A from x_A such that these values may be used independently (in [1], $S_A = x_A D_A$) in session key construction.

4 Certificateless Key Agreement Using Separate KGCs

It may in many cases be desired by users of different KGCs to establish shared keys. For example, in order for encrypted VoIP to be able to operate globally, key agreement and compatibility between networks become a necessary requirement. The following protocol enables session key establishment between users of different KGCs in the certificateless setting.

Setup: Let H, H_1, and H_2 be defined as in the previous protocol. Two different key generation centers, KGC_1 and KGC_2, then respectively generate a key pair $(P_1 = s_1 P \in \mathbb{G}_1, s_1 \in \mathbb{Z}_q^*)$ and $(P_2 = s_2 P \in \mathbb{G}_1, s_2 \in \mathbb{Z}_q^*)$ where P and $\mathbb{G}_1$ are globally agreed. Both KGCs publish their respective public parameters.

Private Key Extraction: This step is similar to the previous protocol, except that entities A and B have their partial private keys generated by different KGCs. Thus, A, under KGC_1, has the private key $S_A = \langle D_A = s_1 Q_A, x_A \rangle$, while B, under KGC_2, has the private key $S_B = \langle D_B = s_2 Q_B, x_B \rangle$.

Key Agreement: A and B generate short-term keys and exchange messages as in the previous protocol. In computing the session key, however, each entity uses the KGC public key of the peer. Thus, A uses $P_2 = s_2P$ in computing the shared key, while B uses $P_1 = s_1P$. A and B then compute respectively:

$$K_A = e(Q_B, P_2 + P_B)^a \cdot e(D_A + x_A Q_A, T_B)$$
$$K_B = e(D_B + x_B Q_B, T_A) \cdot e(Q_A, P_1 + P_A)^b$$

The scheme is consistent because:

$$K = K_A = K_B = e(Q_B, P)^{a(s_2 + x_B)} \cdot e(Q_A, P)^{b(s_1 + x_A)}$$

The final session key is given by $k = H_1(K \| abP \| x_A x_B P)$.

5 Security Attributes

In this section, we will show that the proposed protocol achieves the security attributes identified in Section 2.3. Note that we only consider the basic protocol of Section 3, as it is very similar to the multi-KGC protocol. Furthermore, we trust the KGC not to replace any long-term public keys as, in doing so, a man-in-the-middle attack is made possible (see [1, 5] for discussion).

- **Known session key security**. As short-term keys are used in generating session keys, a compromised session key does not compromise past or future sessions. All protocol runs, even when its participants remain the same, produce a different session key.
- **Forward secrecy**. We let this property constitute two separate parts; both to capture the forward secrecy against an adversary who holds both A and B's long-term private keys (*user forward secrecy*) and against an adversary who has the KGC master key (*KGC forward secrecy*).
  - *User forward secrecy.* Compromising the long-term private keys of entities A and B will not reveal previously established session keys. In order to compute abP of $H_1(K \| abP \| x_A x_B P)$, an adversary must know at least one short-term private key of a given session.
  - *KGC forward secrecy.* Compromise of the KGC master key s does not enable an adversary to reveal previously established session keys. Although the adversary may generate partial private keys, both a short-term private key and the long-term (full) private key of a party involved in a session must be obtained in order to compute the established key.
- **Key-compromise impersonation**. The proposed protocol is resistant to key-compromise impersonation. Assume that the adversary E knows A's private key $S_A = \langle D_A, x_A \rangle$. If E is to impersonate B in a protocol run with A, then E must be able to correctly compute $K = K_A = K_B$. E cannot compute $K_A = e(Q_B, P_B + P_0)^a \cdot e(D_A + x_A Q_A, T_B)$ where $P_0 = sP$ because she does not know the short-term key a. If E was to compute K_B, she would need to know B's long-term private key S_B. Although E could possibly replace B's public key P_B with a value of her choice, she would still need to know $D_B = sQ_B$ in order to successfully impersonate B to A.

- **Unknown key-share**. Suppose an adversary E attempts to make A believe a key is shared with B, while B instead believes the key is shared with E. For E to launch this attack successfully, she should force A and B to share the same secret $K = K_A = K_B$. However, A and B can never share the same key if they don't believe they are mutually communicating. This stems from the fact that both parties use the identifier of the intended peer (i.e. A uses Q_B, while B uses Q_E) in computing the session key. Thus, A cannot verify the authentication tag τ generated by B (passed on by E) and the attack fails.
- **Key control**. Neither party can control the outcome of the session key. However, if A sends her short-term key first, B may be able to predict some bits of the final key by trying different short-term keys before sending the key back to A. Precisely, in computing the shared session key $f(a, b)$ where a is known, B may compute 2^s variants of b and thus select approximately s bits of the joint key. This deficiency exists in all interactive key agreement protocols as pointed out by [6].
- **Known session-specific temporary information security**. Compromising the short-term private keys of a session does not reveal the established key. Specifically, obtaining the keys a and b in any session between entities A and B, allows the adversary to compute $K = (Q_B, P_B + P_0)^a \cdot (Q_A, P_A + P_0)^b$ and abP. However, in order to compute $x_A x_B P$, the adversary must also know at least one long-term private key (or solve the CDH problem). Note that an adversary who is able to obtain short-term private keys is considered very powerful and can break many existing protocols (see [4] for examples).

6 Efficiency

The efficiency of key agreement protocols is essentially measured by the computational and communication overhead. Communication overhead refers to the number of bits transmitted by each entity in a protocol run, while computational overhead refers to the cost of all arithmetic computations each entity must perform in order to carry out the key agreement. Table 1 compares the efficiency of the proposed protocol to the previously proposed protocol by Al-Riyami and Paterson. In evaluating the computational overhead, only heavy operations such as pairings, point multiplications, and pairing exponentiations are considered. Generally, point multiplications and pairing exponentiations are much faster to compute than pairings.

In AP's protocol, each entity is required to exchange three group elements, of which one element represents the short-term public key and the other elements represent the long-term public key. Each entity also must compute four pairings, perform two point multiplications, and make one pairing exponentiation. Note that an entity must still compute four pairings, even when values are *precomputed* (aspects of the session key is computed before a protocol run).

In the proposed protocol, each entity is only required to compute two pairings, perform three point multiplications, and make one pairing exponentiation.

Table 1. Efficiency of certificateless authenticated key agreement protocols

	Al-Riyami-Paterson[1]	Proposed protocol
Message	3 elements of $\mathbb{G}_1$	2 elements of $\mathbb{G}_1$
No precomputation	$4\mathsf{p} + 2\mathsf{m} + 1\mathsf{e}$	$2\mathsf{p} + 3\mathsf{m} + 1\mathsf{e}$
Precomputation	$4\mathsf{p} + 1\mathsf{m}$	$2\mathsf{p} + 2\mathsf{m}$

Notation: (p)airing, point (m)ultiplication, pairing (e)xponentiation

Moreover, the long-term public keys only comprise one group element, and thus, the protocol can be considered more efficient than AP's in terms of computation and message bandwidth. Although the proposed protocol introduces one additional pass and requires entities to compute MACs, AP's protocol should use a similar method in order to prevent an adversary from impersonating parties.

Once long-term public keys have been exchanged in both protocols, only a single pairing is required by the participating parties. Thus, certificateless protocols can be just as efficient as identity-based schemes.

7 Conclusion

This paper has proposed a certificateless authenticated two-party key agreement protocol that is more efficient than the previously proposed protocol by Al-Riyami and Paterson. The protocol is more efficient both in terms of computation and message bandwidth. The protocol also achieves the security attributes that are desired in authenticated key agreement. Furthermore, the proposed protocol can be used to establish keys between users of different key generation centers.

References

[1] Al-Riyami, S.S., Paterson, K.: Certificateless Public Key Cryptography. In: Laih, C.-S. (ed.) ASIACRYPT 2003. LNCS, vol. 2894, pp. 452–473. Springer, Heidelberg (2003)

[2] Blake-Wilson, S., Menezes, A.: Authenticated Diffie-Hellman Key Agreement Protocols. In: Tavares, S., Meijer, H. (eds.) SAC 1998. LNCS, vol. 1556, pp. 339–361. Springer, Heidelberg (1998)

[3] Boneh, D., Franklin, M.: Identity-Based Encryption from the Weil Pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)

[4] Cheng, Z., Nistazakis, M., Comley, R., Vasiu, L.: On The Indistinguishability-Based Security Model of Key Agreement Protocols - Simple Cases. Technical Track Proceedings, Journal of China Information Security. ICISA Press (2004)

[5] Mandt, T.K.: Certificateless Authenticated Two-Party Key Agreement Protocols. Master's Thesis. Gjøvik University College (2006)

[6] Mitchell, C.J., Ward, M., Wilson, P.: Key Control in Key Agreement Protocols. Electronics Letters 34, 980–981 (1998)

[7] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)