Workshop on Cryptology and Information Security

An Efficient MAKEP for Wireless Network*

Chou-Chen Yang† Min-Shiang Hwang‡
Jian-Wei Li† Ting-Yi Chang †

Department of Information and Communication Engineering†
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.
Tel: (886)-4-23323000 ext 4226; Fax: (886)-4-23742375
Email: ccyang@cyut.edu.tw

Department of Information Management‡
Chaoyang University of Technology
168 Gifeng E. Rd., Wufeng,
Taichung County, Taiwan 413, R.O.C.
Email: mshwang@cyut.edu.tw

July 21, 2002

Abstract

In Mutual Authentication and Key Exchange Protocols (MAKEP), public-key-based schemes and symmetric-key-based schemes are the two kinds of schemes most commonly used. However, the former kind has very high computation complexity, and hence it is not suitable for applications in wireless network systems. The later kind, on the other hand, has its limits too. In 2001, Wong et al. proposed the Linear MAKEP. It uses the pre-computation technique to reduce the computation complexity of the wireless device. However, in their scheme, there are too many pairs of private keys to be stored in the MH's memory. In this paper, we shall improve their scheme and store only one pair of private keys instead; at the same time, our improved method will be able to withstand the unknown key-share attack.
Keywords: MAKEP, Low power, pre-computation, unknown key-share attack

*This research was partially supported by the National Science Council, Taiwan, R.O.C., under contract no.: NSC90-2626-E-324-001.
†Responsible for correspondence: Prof. Chou-Chen Yang.

An Efficient MAKEP for Wireless Network

Abstract

In Mutual Authentication and Key Exchange Protocols (MAKEP), public-key-based schemes and symmetric-key-based schemes are the two kinds of schemes most commonly used. However, the former kind has very high computation complexity, and hence it is not suitable for applications in wireless network systems. The later kind, on the other hand, has its limits too. In 2001, Wong et al. proposed the Linear MAKEP. It uses the pre-computation technique to reduce the computation complexity of the wireless device. However, in their scheme, there are too many pairs of private keys to be stored in the MH's memory. In this paper, we shall improve their scheme and store only one pair of private keys instead; at the same time, our improved method will be able to withstand the unknown key-share attack.

Keywords: MAKEP, Low power, pre-computation, unknown key-share attack

1 Introduction

The first public-key-based Mutual Authentication and Key Exchange Protocols (MAKEP) were proposed in [2, 3, 6, 7]. The protocols are very suitable for the general network systems. Unfortunately, they cannot seem to live up to the high standards of the wireless network, where absolutely secure communication must be achieved between the low-power wireless device (we will call it the Mobile Host (MH) in this paper) and the powerful base station (the $Server$). Since the public-key-based MAKEP system has very high computation complexity while the low-power wireless device (MH) depends only on a battery, the MH power will be used up very quickly. The symmetric-key-based schemes [4, 5, 9, 10, 11], on the other hand, were proposed to be more suitable for the wireless network; however, there are two major limits these schemes cannot break through. One is that two parties in communication need to share a long-life key, which means each party must maintain many distinct keys for different parties to get in touch with; the other is that a third trusted party must be involved.

Recently, several schemes [8, 13, 14] have been proposed, without such high computation complexity as the public-key-based scheme has and such restrictions as the symmetric-key-based scheme has. They use a pre-computatin technique to reduce the computation complexity of the MH; in other words, the MH must store some pre-computation result to relieve itself from complex computations. However, in [8, 13], the pre-computation results are on the side the base station ($Server$), so if the MH moves into the realm of another base station ($Server$) and has not the pre-computation result based on the public key of the new $Server$, the MH will not be able to perform these protocols. In another scheme, Wong et al. [14] proposed the Linear MAKEP. this scheme is free from the restriction above, but the MH needs to store n pairs of private keys and n certificates in its storage, which is both unscalable and insecure. In addition, this scheme is vulnerable to the unknown key-share attack [1, 12].

The Linear MAKEP system was then proposed to suit the wireless network more. Such a system can indeed reduce the computation complexity of the MH and thus save the MH's energy of the battery. Yet, in terms of storage space, the MH is not optimized and still has to keep n pairs of private keys. In this paper, we shall propose and improved bersion of the Linear MAKEP, where the MH only keeps one pair of private keys to reduce the storage load. In addition, our new scheme can withstand the unknown key-share attack. In the next section, we will briefly review the Linear MAKEP and show the weakness; then, in section 3, we shall illustrate how our proposed scheme will work in detail; after that, in section 4, the security and performance analysis will be presented; finally, in the last section we shall offer our conclusion.

2 Related work on MAKEP

The following notations are to be used throughout in this article.

- E_K: The encryption transformations under the symmetric key K respectively.
- PK_A: The public key of each entity A under the public key cryptosystem.
- SK_A: Each entity A has a private key under the public key cryptosystem.
- E_{PK_A}: The encryption transformations under the public key cryptosystem.
- Sig_{TA}: A secret signing algorithm of a trusted authority (TA).
- $Cert_A =< ID_A, m, Sig_{TA}(ID_A, m) >$: A certificate $Cert_A$ of the entity A, where ID_A is the identification information of A and m is some message certified by TA that binds to ID_A.
- $r \leftarrow (0,1)^l$: A nonce which is an l-bit random number generated by some cryptographically strong random number.
- k: A security parameter.

In order to achieve secure communication between a low-power wireless device (MH) and a powerful base station ($Server$) under different system requirements, Wong et al. proposed the Linear MAKEP. Compared with previous schemes [2, 3], Wong et al. reduced the load on the computation complexity of the MH while still maintaining a required level of security.

Before running the Linear MAKEP, the MH uses the pre-computation technique. Firstly, the MH must choose a prime p such that the discrete logarithm problem (DLP) in Z_p is intractable, and then the MH chooses a primitive element $g \in Z_p^*$. Then, the MH randomly chooses a sequence of integers $(a_1,\, a_2)$, $(a_3,\, a_4)$, $\cdots$, $(a_{2i-1},\, a_{2i})$ in Z_{p-1} as its private keys, where i is the number of times the MH wants to run the protocol. The corresponding sequence of public key pairs are $(g^{a_1},\, g^{a_2})$, $(g^{a_1},\, g^{a_2})$, $\ldots$, $(g^{a_{2i-1}}, g^{a_{2i}})$ in Z_p, where $1 \leq i$. Secondly, the signatures $Sig_{TA}(ID_{MH}, g^{a_{2i-1}}, g^{a_{2i}})_{1 \leq i}$ are obtained from the TA.

The protocol runs as follows:

(1) $MH \rightarrow Server$: $Cert^i_{MH}$

At the i-th run of the protocol, the MH constructs a certificate denoted by $Cert^i_{MH} = \langle ID_{MH},\, g^{a_{2i-1}},\, g^{a_{2i}},\, Sig_{TA}(ID_{MH},\, g^{a_{2i-1}},\, g^{a_2 i}) \rangle$ and sends it to the $Server$.

(2) $Server \rightarrow MH$: r_S

Upon receiving (1), the $Server$ confirms the validity of the certificate and sends back a nonce r_S.

(3) *MH: Upon receipt of (2).*

The MH chooses another nonce r_{MH} and computes $x = E_{PK_S}(r_{MH})$. Then it computes y as

$$y = a_{2i-1}(x \oplus r_S) + a_{2i} \bmod (p-1). \qquad (I)$$

(4) $MH \rightarrow Server$: x, y

The MH computes a new session key σ as $r_{MH} \oplus y$.

(5) *$Server$: Upon receipt of (4).*

The $Server$ checks the equation

$$(g^{a_{2i-1}})^{(x \oplus r_S)} \cdot g^{a_{2i}} \stackrel{?}{\equiv} g^y \bmod p.$$

If the equation holds, the $Server$ derives r_{MH} by decrypting x and then computes a new session key σ as $r_{MH} \oplus y$; otherwise, this communication is rejected and the protocol halts.

(6) $Server \rightarrow MH$: $E_\sigma(x)$

As soon as the MH receives the message $E_\sigma(x)$, it decrypts the message and then checks whether the decrypted message is x.

MH		$Server$
$(a_1, a_2, \cdots, a_{2n}) \in_R Z_{p-1}$		(PK_S, SK_S)
$(g^{a_1}, g_2^{a}, \ldots, g^{a_{2n}}) \in Z_p^*$		
	(1) $Cert^i_{MH}$ $\longrightarrow$	
		$r_S \in_R Z_{P-1}$
	$\longleftarrow$ (2) r_S	
(3) $r_{MH} \leftarrow \{0,1\}^k$		
$x = E_{PK_S}(r_{MH})$		
$y = a_{2i-1}(x \oplus r_S) + a_{2i} \bmod (p-1)$		
	(4) x, y $\longrightarrow$	
$\sigma = r_{MH} \oplus y$		(5) $(g^{a_{2i-1}})^{(x \oplus r_S)} \cdot g^{a_{2i}} \stackrel{?}{\equiv} g^y (\mathrm{mod} p)$
		$\sigma = r_{MH} \oplus y$
	$\longleftarrow$ (6) $E_\sigma(x)$	

Figure 1: The Linear MAKEP

Since the MH uses the pre-computation technique to reduce its computation demand, it must use a larger storage space to store these pre-computation results including one prime p, n pairs of private keys and n certificates. However, the private keys are in person of the MH, the MH must take more effort to maintain n pairs of private keys compared if one pair of private key. Hence we will propose that the MH only own one pair of private keys.

Secondly, the private keys are randomly chosen integers, and hence the values of the private keys may be the same. It will cause an attacker to be more likely to obtain the private keys by equation (I) if she/he has collected communication messages from all the running protocols. For example, assume an attacker has the communication messages of the u-th as well as the v-th running protocol (y, x and r_S), where the private keys are as follows:

$a_{2u-1} = a_{2v-1} = T$; $a_{2u} = a_{2v} = Q$;

According to equation (I), the attacker owns the two following equations:

$y_u = a_{2u-1}(x_u \oplus r_{S_u}) + a_{2u} \bmod (p-1)$

$y_v = a_{2v-1}(x_v \oplus r_{S_v}) + a_{2v} \bmod (p-1)$

These equations can also be presented as

$y_u = T(x_u \oplus r_{S_u}) + Q \bmod (p-1)$

$y_v = T(x_v \oplus r_{S_v}) + Q \bmod (p-1)$

Because the attacker knows y, x and r_S, she/he can derive T and Q. Then she/he can use the private key and the corresponding certificates to impersonate the MH.

MH		E		$Server$
$(a_1, a_2, \cdots, a_{2n}) \in_R Z_{p-1}$ $(g^{a_1}, g^a_2, \ldots, g^{a_{2n}}) \in Z_p^*$		$c \in_R Z_{p-1}$		(PK_S, SK_S)
	$\xrightarrow{(1)\ Cert^i_{MH}}$		$\xrightarrow{(1')\ Cert^i_E}$	
				$r_S \in_R Z_{P-1}$
	$\xleftarrow{(2)\ r_S}$		$\xleftarrow{(2)\ r_S}$	
(3) $r_{MH} \leftarrow \{0,1\}^k$ $x = E_{PK_S}(r_{MH})$ $y = a_{2i-1}(x \oplus r_S) + a_{2i} \bmod (p-1)$				
	$\xrightarrow{(4)\ x, y}$		$\xrightarrow{(4')\ x, y' = yc}$	
$\sigma = r_{MH} \oplus y$				(5) $(g^{c \cdot a_{2i-1}})^{(x \oplus r_S)} \cdot g^{c \cdot a_{2i}} \overset{?}{\equiv} g^{y'} (\mathrm{mod} p)$ $\sigma = r_{MH} \oplus y$
	$\xleftarrow{(6)\ E_\sigma(x)}$		$\xleftarrow{(6)\ E_\sigma(x)}$	

Figure 2: The unknown key-share attack on the Linear MAKEP

Lastly, the Linear MAKEP is vulnerable to the unknown key-share attack [1, 12]. Before the description of the unknown key-share attack on the Linear MAKEP, we assume that an adversary E selects an integer $c \in Z^*_{p-1}$, computes its public key as $PK_E = (g^{a_{2i}})^c$, and gets its certificate denoted by

$Cert^i_E = \langle ID_E, (g^{a_{2i-1}})^c, (g^{a_{2i}})^c, Sig_{TA}(ID_E, (g^{a_{2i-1}})^c, (g^{a_{2i}})^c)\rangle$.

The Linear MAKEP is shown to be insecure against the unknown key-share attack in Figure 2. The attack is executed as follows.

(1) $MH \rightarrow Server$: $Cert^i_{MH}$

(1′) $E \rightarrow Server$: $Cert^i_E$

An adversary E intercepts MH's ephemeral public information $Cert^i_{MH}$ and replaces it with $Cert^i_E$

(2) $Server \rightarrow MH$: r_S

Upon receipt of (1′), the $Server$ sends back a nonce r_S after verifying $Cert^i_E$.

(3) *MH: Upon receipt of (2).*

The MH chooses another nonce r_{MH} and computes $x = E_{PK_S}(r_{MH})$. Then it computes y as

$y = a_{2i-1}(x \oplus r_S) + a_{2i} \bmod (p-1)$.

(4) $MH \rightarrow Server$: x, y

The MH computes a new session key σ as $r_{MH} \oplus y$.

(4′) E intercepts x, y and computes $y' = yc$, and then E transmits y' to the $Server$.

(5) *Server: Upon receipt of (4').*

The $Server$ checks the equation

$(g^{c \cdot a_{2i-1}})^{(x \oplus r_S)} \cdot g^{c \cdot a_{2i}} \stackrel{?}{\equiv} g^{y'} \bmod p$ by using E's public key. Indeed, it holds, beacuse $(g^{c \cdot a_{2i-1}})^{(x \oplus r_S)} \cdot g^{c \cdot a_{2i}} = g^{(c \cdot a_{2i-1})(x \oplus r_S)+(c \cdot a_{2i})} = g^{yc} = g^{y'} \bmod p$

And the $Server$ derives r_{MH} by decrypting x and then computes a new session key σ as $r_{MH} \oplus y$. The adversary E cannot retrieve the session key σ. However, the server mistakenly believes that it shares σ with E while the MH believes that it shares σ with the $Server$.

(6) $Server \rightarrow MH$: $E_\sigma(x)$

As soon as the MH receives the message $E_\sigma(x)$, it decrypts the message and then checks whether the decrypted message is x. At the same time, the MH believes that it has shared σ with the $Server$.

In our improved scheme, we shall prevent the unknown key-share attack from faking effect by replacing the session key $\sigma= r_{MH} \oplus y$ with $\sigma = r_{MH} \oplus y||ID_{MH}$. If the $Server$ mistakenly believes that it shares σ with E while the MH believes that it shares σ with the $Server$, the session key $\sigma = r_{MH} \oplus y||ID_E$ which the $Server$ computes will not be equal to the session $\sigma = r_{MH} \oplus y||ID_{MH}$ the MH computes. Therefore, our new method can indeed resist the unknown key-share attack.

3 Our proposed Scheme

As happens in the Linear MAKEP [14], the MH also must do some pre-computation operations in our protocol. Firstly, the MH must choose a prime p such that the discrete logarithm problem (DLP) in Z_p is intractable, and then the MH must choose a primitive element $g \in Z_P^*$. However, differently, the MH only chooses two integers (a_1, a_2) in Z_{p-1} as its private key pair, and the corresponding public key pair is g^{a_1} and $g^{a_1 \oplus a_2{}^{2^j}}$ in Z_{p-1}, where j is the j-th round of running the protocol. Secondly, the signatures $Sig_{TA}(ID_{MH}, g^{a_1}, g^{a_1 \oplus a_2{}^{2^j}})_{1 \le j}$ are obtained from the TA.

Our new protocol runs as follows:

(1) $MH \rightarrow Sever$: $Cert^j_{MH}$

At the j-th run of the protocol, the MH constructs a certificate denoted by $Cert^j_{MH}= \langle ID_{MH},\ g^{a_1},\ g^{a_1 \oplus a_2{}^{2^j}},\ Sig_{TA}(ID_{MH}, g^{a_1}, g^{a_1 \oplus a_2{}^{2^j}})\rangle$.

(2) $Sever \rightarrow MH$: r_S

Upon receipt of (1), the Sever confirms the validity of the certificate and sends back a nonce r_S.

(3) *MH: Upon receipt of (2).*

The MH chooses another nonce r_{MH} and computes $x = E_{PK_S}(r_{MH})$. Then it computes y as

$$y = a_1(x \oplus r_S) + a_1 \oplus a_2^{2^j} \bmod (p-1). \qquad (II)$$

Because $a_2^{2^j}$ can be computed as

$$a_2^{2^j} = a_2^{2^{j-1}} \cdot a_2^{2^{j-1}},$$

the MH has stored the $a_2^{2^{j-1}}$ when running the $j-1$-th protocol, and thus it can obtain $a_2^{2^j}$ by multiplying $a_2^{2^{j-1}}$ by $a_2^{2^{j-1}}$ when running the j-th protocol. By doing so, it can reduce its computation demand.

(4) $MH \rightarrow Sever$: x, y

The MH computes a new session key σ as $r_{MH} \oplus y || ID_{MH}$.

(5) *Sever: Upon receipt of (4).*

The $Server$ checks whether

$$(g^{a_1})^{x \oplus r_S} \cdot g^{a_1 \oplus a_2^{2^j}} \stackrel{?}{\equiv} g^y \ (mod\, p). \qquad (III)$$

If so, the $Server$ derives r_{MH} by decrypting x. and then computes a new session key σ as $r_{MH} \oplus y || ID_{MH}$; otherwise, this communication is rejected and the protocol halts.

(6) $Server \rightarrow MH$: $E_\sigma(x)$

As soon as the MH receives the message $E_\sigma(x)$, the MH decrypts the message and then checks whether the decrypted message is x.

Like the Linear MAKEP [14], the mutual authentication is achieved by (r_S, y) and (x, $E_\sigma(x)$), in addition, the r_S and r_{MH} which are not only encrypted by the public key of the Sever but also signed by the MH through a signature-like mechanism in (II) and (III), are bound together to provide the $Server$ and MH with the ability to confirm the freshness of the session keys. In our new scheme, the MH owns only one pair of private keys (a_1 and a_2), and there are corresponding public keys g^{a_1} and $g^{a_1 \oplus a_2^{2^j}}$ required in the j-th run of the

MH $\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad$ $Server$

$(a_1,\, a_2) \in_R Z_{p-1}$ $\qquad\qquad\qquad\qquad\qquad\qquad$ $(PK_S,\, SK_S)$

$(g^{a_1},\, g^{a_1 \oplus a_2{}^{2^j}}) \in Z_p^*$

$$\xrightarrow{\quad (1)\ Cert^i_{MH} \quad}$$

$r_S \in_R Z_{P-1}$

$$\xleftarrow{\quad (2)\ r_S \quad}$$

(3) $r_{MH} \leftarrow \{0,1\}^k$

$x = E_{PK_S}(r_{MH})$

$y = a_1(x \oplus r_S) + a_1 \oplus a_2^{2^j} \mod (p-1)$

$$\xrightarrow{\quad (4)\ x, y \quad}$$

$\sigma = r_{MH} \oplus y||ID_{MH}$ $\qquad\qquad$ (5) $(g^{a_1})^{x \oplus r_S} \cdot g^{a_1 \oplus a_2^{2^j}} \stackrel{?}{\equiv} g^y (\mathrm{mod} p)$

$\sigma = r_{MH} \oplus y||ID_{MH}$

$$\xleftarrow{\quad (6)\ E_\sigma(x) \quad}$$

Figure 3: Our Protocol

protocol.

4 Security and performance analysis

In our new scheme, the private key pair of the MH is $(a_1,\, a_2)$ in Z_{p-1}, and the corresponding public key pair is g^{a_1} and $g^{a_1 \oplus a_2{}^{2^j}}$ in Z_{p-1}, where j is the j-th run of the protocol. Assume the attacker wants to obtain the private key of the MH. One method is to solve (I); however, it is difficult for the attacker to obtain $(a_1,\, a_2)$, because our scheme depends on the the difficulty of finding the composite exclusive-OR operation and solving the discrete logarithm problem. That mean the attacker is not likely to obtain the MH's private key pair. To prevent the unknown key-share attack, we use $r_{MH} \oplus y||ID_{MH}$ as the session key σ. Take an example. If the unknown key-share attack happens, the session key $\sigma = r_{MH} \oplus y||ID_E$ which the $Server$ computes will not be equal to the session $\sigma = r_{MH} \oplus y||ID_{MH}$ which the MH computes. Therefore, the method can prevent the unknown key-share attack from causing any damage.

As for the performance of our new scheme, although the total computation complexity of the MH in our protocol must is one module multiplication and one exclusive-OR operation more than that of the Linear MAKEP, we can protect the private key pair of the MH from being stolen and only maintain this one pair private keys instead of n pairs. Besides, our proposed scheme is capable of preventing the unknown key-share attack.

5 Conclusion

In this paper, we have modified in the Linear MAKEP to make it more efficient and powerful. After the security and performance analysis, we have demonstrated that the storage consumed by the MH in our protocol is less than that of the Linear MAKEP. Although the total computation complexity of the MH in our protocol is one module multiplication and one exclusive-OR operation more than the Linear MAKEP, the attacker will not obtain the MH's private keys by intercepting many communication messages. In addition, our proposed scheme is capable of preventing the unknown key-share attack.

References

[1] "Cryptanalysis of Mutually Authenticated Key Exchange Protocol for Mobile Devices," *to appear in International Journal of Asiacrypt2002*, 2002.

[2] P. A. O., Karlton and P. C. Kocher, "The SSL Protocol Version 3.0," *Internet-Draft*, Nov 1996.

[3] A. Azziz and W Diffie, "A Secure Communications Protocol to Prevent Unauthorized Access - Privacy and Authentication for Wireless Local Area Networks," *IEEE Personal Communications*, (First Quarter 1994).

[4] M. Bellare and P. Rogaway, "Provably Secure Session Key Distribution the Three Party Case," *Proceedings of the 27th ACM Symposium on the Theory of Computing*, 1995.

[5] M. Bellare and P. Rogaway, "Entity Authentication and Key Distribution," *In Douglas R. Stinson, editor, Advances in Cryptology – CRYPTO '93*, vol. volume 773 of Lecture Notes in Computer Science, pp. 232–249, 22-26 August 1993.

[6] Simon Blake-Wilson and Alfred Menezes, "Authenticated Diffie-Hellman key agreement protocols," *In 5th annual international workshop, SAC'98*, no. 1556, pp. 339–361, Springer-Verlag, 1998.

[7] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, pp. 644–654, Nov. 1976.

[8] M. Jacobsson and D. Poincheval, "Mutually authentication for low power mobile device," *Proceeding of Finaltial Cryptography*, no. Springer-Verlag, February 2001.

[9] J. Kohl and C. Neuman, "The Kerberos Network Authentication Service (V5)," *IETF RFC1510*, Sep 1993.

[10] R. M. Needham and M. D. Schroeder, "Using Encryption for Authentication in Large Networks of Computers," *Communication of the ACM*, pp. 993–999, Dec. 1978.

[11] Dave Otway and Owen Rees, "Efficient and timely mutual authentication," *Operating Systems Review*, Jan 1987.

[12] S.Blake-Wilson and A.Menezes, "Unknown key-share Attacks on the station-to-station (sts) protocol," *Public Key Cryptography – Proceedings of PKC '99*, vol. 1560, pp. 154–170, 1999.

[13] D. S. Wong and A. H. Chan, "Efficient and mutually authenticated key exchage for low power mobile device," *Advances in cryptology; Asiacypt'01*, no. Springer-Verlag, pp. 272–289, 2001.

[14] D. S. Wong and A. H. Chan, "Mutual Authentication and Key Excahge for Low Power Wireless Communications," *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force. IEEE*, vol. 1, pp. 39–43, 2001.