A Secure and Efficient Key Exchange Protocol for Mobile Communications

Fuw-Yi Yang and Jinn-Ke Jan*

Department of Applied Mathematics, National Chung Hsing University,
Taichung 402, Taiwan, R.O.C., Email: yangfy@ms7.hinet.net

*Department of Computer Science, National Chung Hsing University,
Taichung 402, Taiwan, R.O.C., Email: jkjan@cs.nchu.edu.tw

Abstract

This paper proposes a key exchange protocol with mutual authentication, which requires only 0.1 modular multiplications for online computations. This online computation is ten times faster than that of conventional protocols. The message size of the proposed protocol is about half (50%~66%) that of the previous protocols. In addition to its efficiency in online computation and bandwidth, the paper provides a formal proof to guarantee the security of the proposed protocol. Possessing of both secure and efficient properties makes the proposed protocol suitable for the low power mobile communications.

1. Introduction

The L-MAKEP (linear mutual authentication key exchange protocol) proposed in [1] is a key exchange protocol specially designed for the wireless communications between a low power mobile device (client) and a powerful base station (server). The goal of the L-MAKEP is to make the computational complexity as efficient as possible at the client end without decreasing the strength of security. Consequently, the server sides do most of the computations. It is a good idea in the environment of the wireless communications, especially in the case of a lower power client end.

However, there are flaws in L-MAKEP. The scheme in [2] successfully mounted an unknown key-share attack [3-6] on this protocol and proposes an improved protocol.

A scenario of unknown key-share attack proposed in [3-4] is as follows. Assume that an account holder *U* (user *U*) initiates a session communication with a bank server *S*, and an adversary *E* mounts an unknown key share attack on this session. As a result of the attack, user *U* believes that a session key is shared with the server *S* and server *S* believes that the session key is shared with the adversary *E*. Although adversary *E* does not obtain the session key, this attack could cause problem. For example, if user *U* initiates the protocol to deposit an electronic fund in his account, the deposit will eventually be made to the adversary *E*'s account. The protocol L-MAKEP and its improvement in [2] are also insecure under the attack of middle-man as shown by the scheme in [7]. The work in [7] also proposes an improved protocol to resist the attacks of middle-man and unknown key-share; we call the protocol I-MAKEP.

This paper proposes an efficient and secure key exchange protocol for mobile communications. Although both participants are authenticated through the public key cryptography, the online computational cost for client side is only *0.1* modular multiplications. This computational complexity is ten times faster than that of the protocols L-MAKEP and I-MAKEP. The message size for the proposed protocol is *3712* bits while the message size for the protocol L-MAKEP is *7328* bits and for the protocol I-MAKEP is *5600* bits (Section 3 will show the details.). Moreover, the proposed protocol consists of only three messages while L-MAKEP and I-MAKEP consist of four messages (two round trips). The savings in computations, bandwidth, and messages is valuable for key exchange protocols, especially in the environment of wireless communications with low power mobile devices. In addition, the paper provides a formal proof to show that breaks the security of the proposed protocol is to solve the difficulty of the factoring problem. Thus the proposed protocol not only resists to the attacks mentioned above but also resists to other potential attacks.

The organization of this paper is as follows. Section 2 presents the new protocol and describes the notation used in the paper. Section 3 compares the computational cost and message size between the proposed protocol, L-MAPEP, and I-MAKEP. In Section 4, the security of the new protocol is investigated and proven. Finally, Section 5 concludes the paper.

2. Protocol proposed

This section presents an efficient and secure key exchange protocol with mutual authentication (ES-MAKEP). The protocol is inspired by the concept of chameleon hash function in [8]. We first describe the cryptographic settings for the protocol.

Let $\varepsilon_{PK}()$ denote an asymmetric encryption function and $\delta_{SK}()$ be the corresponding decryption function. Similarly, $E_K()$ represents a symmetric encryption function and $D_K()$ is the decryption function so that $m = D_K(E_K(m))$. Both the encryption functions $\varepsilon_{PK}()$ and $E_K()$ are assumed to be secure in the model of adaptive chosen ciphertext attack [9-11]. In this attack model, an adversary is given a decryption oracle to decrypt the ciphertexts adaptively chosen by the adversary except the one that he is trying to decrypt. The identification of a client entity U (user U) is denoted by ID_U and the identification of a server S is represented by ID_S. Server S has a private key SK_S and the corresponding public key PK_S. The public key and secret key of user U are selected in the following way. User U randomly chooses four large prime numbers p, q, p', and q' such that $p = 2\,p' + 1$ and $q = 2\,q' + 1$. Then, he or she randomly selects an element g of order $\lambda(n)$ from the multiplicative group Z_n^*, where $n = p\,q$ and $\lambda(n) = lcm(p - 1, q - 1)$. Thus user U has the private key (p, q) and the public key (g, n). In addition, $x \,||\, y$ denotes that string x concatenates string y, $|n|$ represents bit length of n, $r \in_R G$ denotes that r is a random number selected from the set G, and l indicates the length of session keys, *e.g. 160* bits for a typical value suggested in [12]. The following steps describe the details of the protocol ES-MAKEP.

1. User U randomly selects three numbers r_{UK}, r_{UR} and r_{UF}, *i.e.* $r_{UK}, r_{UF} \in_R \{0, 1\}^l$ and $r_{UR} \in_R Z_{\lambda(n)}$. Compute the quantities $C1_{r_{UK}} = \varepsilon_{PK_S}(r_{UK})$ and $CMT = g^{r_{UF} \| r_{UR}} \bmod n$, *i.e.* encrypt the random number r_{UK} and calculate the commitment of r_{UF} and r_{UR}. Then, user U sends server S message $M1 = \{C1_{r_{UK}}, CMT, ID_U\}$ to ask for initiating a new session.
2. On receiving the message $M1$, server S decrypts the ciphertext $C1_{r_{UK}}$ to obtain r_{UK}. S also selects a random numbers $r_{SK} \in_R \{0, 1\}^l$, calculates the session key $\sigma_{SU} = r_{SK}$

$\oplus$ r_{UK}, and encrypts the random number r_{UK}, *i.e.* $C2_{r_{UK}} = E_{\sigma_{SU}}(r_{UK})$. Then, server *S* sends $M2 = \{r_{SK}, C2_{r_{UK}}\}$ to challenge user *U*.

3. Upon receiving the challenge message *M2*, *U* calculates the session key $\sigma_{US} = r_{UK} \oplus r_{SK}$ and decrypts the ciphertext $C2_{r_{UK}}$ to obtain a random number r'_{UK}, *i.e.* $r'_{UK} = D_{\sigma_{US}}(C2_{r_{UK}}) = D_{\sigma_{US}}(E_{\sigma_{SU}}(r_{UK}))$. Note that σ_{SU} is equal to σ_{US} if messages *M1* and *M2* are successfully transmitted. User *U* authenticates server *S* by checking $r_{UK} = r'_{UK}$. After authenticating server *S*, *U* constructs a response message as follows.

   3.1. Compute the quantities $S_F = h(r_{UK}, r_{SK}, ID_U, ID_S)$ and $C3 = E_{\sigma_{SU}}(ID_U)$.

   3.2. Solve for S_R in equation (1).

$$2^{|n|} r_{UF} + r_{UR} = 2^{|n|} S_F + S_R \bmod \lambda(n)$$

$$S_R = 2^{|n|} (r_{UF} - S_F) + r_{UR} \bmod \lambda(n) \quad (1)$$

   U sends the response message $M3 = \{C3, S_R\}$ to server *S*.

4. *S* computes the quantities $S_F = h(r_{UK}, r_{SK}, ID_U, ID_S)$ and $CMT' = g^{S_F \| S_R} \bmod n$. *S* authenticates user *U* by verifying $CMT = CMT'$.

For easy reading, the message flows are listed below.

$U \rightarrow S$: $M1 = \{C1_{r_{UK}}, CMT, ID_U\}$, where $C1_{r_{UK}} = \varepsilon_{PK_S}(r_{UK})$ and $CMT = g^{r_{UF} \| r_{UR}}$.
$S \rightarrow U$: $M2 = \{r_{SK}, C2_{r_{UK}}\}$, where $C2_{r_{UK}} = E_{\sigma_{SU}}(r_{UK})$ and $\sigma_{SU} = r_{UK} \oplus r_{SK}$.
$U \rightarrow S$: $M3 = \{C3, S_R\}$, where $C3 = E_{\sigma_{SU}}(ID_U)$, $2^{|n|} r_{UF} + r_{UR} = 2^{|n|} S_F + S_R$ and $S_F = h(r_{UK}, r_{SK}, ID_U, ID_S)$.

3. Performance

For each step in the protocols ES-MAKEP, L-MAKEP, and I-MAKEP, the computations and message sizes are displayed in Table 1, 2 and 3. The cost of computations is further divided into two parts: online and offline computations. To simplify the estimation of message sizes and computations, we have assumed that the asymmetric encryption is a *1024*-bit RSA encryption with a low exponent public key $e = 3$, the symmetric encryption outputs a stream of *160*-bit and ID_U is a *160*-bit

encoding. Also, the cost of additions, hash operations, and symmetric encryption and decryption are not included. For a practical cryptographic settings [12], $|n| = 1024$ bits and $|r_{UF}| = |S_F| = 160$ bits, the computational cost (software implementation) for solving S_R is estimated to be only about *0.1* modular multiplications of two *1024*-bit numbers modulo a *1024*-bit modulus [8]**.** Note that computing S_R involves only a reduction of a *1184*-bit number to a *1024*-bit number while computing a modular multiplication of two *1024*-bit numbers requires a conventional multiplication and a reduction of a *2048*-bit number to a *1024*-bit number. For easy comparison, the message flows of protocols L-MAKEP and I-MAKEP are listed below.

The message flows of L-MAKEP: The server *S* has published a public key PK_S, large prime *p* and a primitive element $g \in Z_p^*$.

$U \rightarrow S$: $M1 = \{ID_U, g^{a_{2i-1}}, g^{a_{2i}}, Sig_{TA}(ID_U, g^{a_{2i-1}}, g^{a_{2i}})\}$, where $a_{2i-1}, a_{2i} \in_R Z_{p-1}$, and $Sig_{TA}(m)$ is a signature of a trusted authority (TA) on the message *m*. Server *S* should verify the signature $Sig_{TA}(ID_U, g^{a_{2i-1}}, g^{a_{2i}})$.

$S \rightarrow U$: $M2 = \{r_S\}$, where $r_S \in_R Z_{p-1}$.

$U \rightarrow S$: $M3 = \{x, y\}$, where $x = \varepsilon_{PK_S}(r_U)$, $y = (a_{2i-1}(x \oplus r_S) + a_{2i}) \bmod (p-1)$, and $r_U \in_R \{0, 1\}^{160}$. User *U* computes the response message *M3* and server *S* verifies it by checking $g^y = g^{a_{2i-1}(x \oplus r_S)} \; g^{a_{2i}} \bmod p$.

$S \rightarrow U$: $M4 = \{E_\sigma(x)\}$, where $\sigma = r_U \oplus y$.

The message flows of I-MAKEP: The server *S* has private key *(d, p, q)* and public key *(e, g, N)*, where *p, q* are two large primes, $N = p \times q$, *g* has maximum order in the group Z_N^*, $\phi(N) = (p-1)(q-1)$, $gcd(e, \phi(N)) = 1$, and $e \times d = 1 \bmod \phi(N)$. The user *U* uses the pair (ID_U, v) to register at server *S* and receives a certificate $y = (v - ID_U)^d \bmod N$, where $v = g^{-x} \bmod N$ and $x \in_R Z_N$ is the secret key of *U*.

$U \rightarrow S$: $M1 = \{ID_U, y\}$. Server *S* verifies *M1* by checking $v = y^e + ID_U \bmod N$.

$S \rightarrow U$: $M2 = \{r_S\}$, where $r_S \in_R Z_N$.

$U \rightarrow S$: $M3 = \{u, t, s\}$, where $u = g^w \bmod N$, $t = \varepsilon_{PK_S}(k)$, $s = w + x\,h(r_S, t, u)$, and *w* and $k \in_R Z_N$. User *U* computes the response message *M3* and server *S* verifies it

by checking $u = v^{h(r_S,t,u)} g^s \bmod N$.

S→U: M4 = {h(k)}.

As shown in Table 1, the client's online computations for ES-MAKEP, L-MAKEP, and I-MAKEP are *0.1*, *1*, and *1* MMs, respectively. Although L-MAKEP requires only two MMs for offline computation, this protocol requires a trusted authority to issue several commitments, which are the sources of the middle-man attack described in [7]. Table 2 and Table 3 show that among the three protocols, ES-MAKEP has the most efficient in computations and the least requirement in bandwidth and messages.

Table 1: Client's computations in protocols ES-MAKEP, L-MAKEP, I-MAKEP

	ES-MAPEP		L-MAKEP		I-MAKEP	
	Online	Offline	Online	Offline	Online	Offline
Message *M1*	*0*	*1778* MMs[a]	*0*	*0*	*0*	*0*
Message *M2*	*0*	*0*	*0*	*0*	*0*	*0*
Message *M3*	*0.1* MMs	*0*	*1* MM	*2* MMs[b]	*1* MM	*1538* MMs[c]
Message *M4*	*None*	*None*	*0*	*0*	*0*	*0*
Total	*0.1* MMs	*1778* MMs	*1* MM	*2* MMs	*1* MM	*1538* MMs

a. Computing $C1_{r_{UK}} = \varepsilon_{PK_S}(r_{UK})$ requires *2* MMs and computing $CMT = g^{r_{UF} \| r_{UR}}$ requires *1.5 * (160 + 1024) = 1776* MMs, where MM denotes modular multiplications of two *1024*-bit numbers modulo *1024*-bit modulus.

b. Encryption requires *2* MMs (assume RSA encryption and public key $PK_S = 3$).

c. Computations of commitment requires *1.5 * 1024 = 1536* MMs and encryption requires *2* MMs (public key $e = 3$).

Table 2: Server's computations in protocols ES-MAKEP, L-MAKEP, I-MAKEP

	ES-MAKEP		L-MAKEP		I-MAKEP	
	Online	Offline	Online	Offline	Online	Offline
Message *M1*	*1536* MMs	*0*	*2* MMs[d]	*0*	*2* MMs[e]	*0*
Message *M2*	*0*	*0*	*0*	*0*	*0*	*0*
Message *M3*	*1776* MMs[f]	*0*	*3456* MMs[g]	*0*	*3333* MMs[h]	*0*
Message *M4*	*None*	*None*	*0*	*0*	*0*	*0*
Total	*3312* MMs	*0*	*3458* MMs	*0*	*3335* MMs	

d. Verification requires *2* MMs (assume TA uses RSA cryptosystem and verification key $e = 3$).

e. Computational cost of encryption requires *2* MMs (public key $e = 3$).

f. Computing $CMT' = g^{S_F \| S_R}$ requires $1.5 * (160 + 1024) = 1776$ MMs.

g. Decryption requires $1.5 * 1024 = 1536$ MMs and verification requires $1.25 * 1.5 * 1024 = 1920$ MMs, by the technique of simultaneous multiple exponentiations [13].

h. Decryption requires $1.5 * 1024 = 1536$ MMs and verification requires $1.17 * 1.5 * 1024 = 1797$ MMs.

Table 3: The message sizes of protocol ES-MAKEP, L-MAKEP, I-MAKEP

	ES-MAKEP	L-MAKEP	I-MAKEP
Message *M1*	*2208* bits	*3232* bits	*1184* bits
Message *M2*	*320* bits	*1024* bits	*1024* bits
Message *M3*	*1184* bits	*2048* bits	*3232* bits
Message *M4*	*none*	*1024* bits	*160* bits
Total	*3712* bits	*7328* bits	*5600* bits

4. Security analysis

In the followings, subsection 4.1 proves that the proposed protocol ES-MAKEP is a mutual authentication protocol (Theorem 1 and 2) and has resistance to those attacks proposed in [2, 7] , subsection 4.2 proves that the protocol is a secure protocol (Lemma 3, 4, and 5).

4.1 ES-MAKEP is a mutual authentication protocol

Theorem 1. Protocol ES-MAKEP correctly authenticates user *U*.

Proof. If user *U* knows the secret key *(p, q)*, then *U* can compute the quantity $\lambda(n) = lcm(p - 1, q - 1)$, solve for S_R in (1) and construct the response message $M3 = \{C3, S_R\}$. Since $2^{|n|} r_{UF} + r_{UR} = 2^{|n|} S_F + S_R \bmod \lambda(n)$, the equation $g^{r_{UF}\|r_{UR}} = g^{S_F\|S_R} \bmod n$ is obtained. Thus server *S* successfully authenticates user *U*.

Assume that an adversary *E* does not now the private key *(p, q)* of user *U* and successfully impersonates the user *U*, then *E* has generated the quantities S_F and S_R such that $g^{r_{UF}\|r_{UR}} = g^{S_F\|S_R} \bmod n$. Thus equation (2) is obtained.

$$g^{r_{UF}\|r_{UR} - S_F\|S_R} = g^{w} = 1 \bmod n \tag{2}$$

Equation (2) implies that *w* is a multiple of $\lambda(n)$. Since $\lambda(n) = lcm(p - 1, q - 1)$ and $\phi(n) = (p - 1)(q - 1)$, then $\phi(n)$ divides $(2w)$. The algorithm in [14] shows that the factorization of *n* can be computed efficiently, if any multiple of $\phi(n)$ is known. The public key *n* thus can be factored by the adversary *E* using the algorithm in [14]. This conclusion contradicts the intractable assumption of factoring problem. Therefore, if server *S* successfully authenticates user *U*, then *U* knows the private key *(p, q)*. □

Theorem 2. Protocol ES-MAKEP correctly authenticates server *S*.

Proof. If server *S* knows the secret key SK_S, then *S* can decrypt the ciphertext $C1_{r_{UK}}$ to obtain r_{UK}, calculate the session key $\sigma_{SU} = r_{UK} \oplus r_{SK}$, and generate the ciphertext $C2_{r_{UK}} = E_{\sigma_{SU}}(r_{UK})$. On receiving r_{SK}, *U* calculates the session key $\sigma_{US} = r_{SK} \oplus r_{UK}$, using the stored r_{UK}. Thus, the session keys σ_{SU} and σ_{US} have the same quantity.

Clearly, the quantity recovered by decrypting the ciphertext $C2_{r_{UK}}$ will be equal to the quantity of stored r_{UK}.

With overwhelming probability, S knows the secret key SK_S, if user U authenticates server S as legal. Namely, only S can decrypt the ciphertext $C1_{r_{UK}}$ to obtain the random number r_{UK}. This result is derived from the security of the encryption functions $\varepsilon_{PK}()$ and $E_K()$, which is assumed to be secure against the adaptive chosen ciphertext attack. Therefore, server S is successfully authenticated by U if and only if S knows the secret key SK_S. □

The attacks described in [2, 7] are performed by modifying messages intercepted from previous sessions. Thus by Theorem 1 and 2, the protocol ES-MAKEP is resistant to them.

4.2 ES-MAKEP is a secure protocol

This subsection investigates the security of the protocol ES-MAKEP. We adopt the security measure and those attack model used in [15-16]. Assume that an adversary with total control over the communication channels can mount parallel attacks, and is told the previous session keys. A key exchange protocol is secure if the following requirements are satisfied.

1. If both participants honestly execute the protocol, then the session key is $\sigma = \sigma_{US} = \sigma_{SU}$.
2. No one can calculate the session key σ except participants U and S.
3. The session key is indistinguishable from a truly random number.

Lemma 3. Protocol ES-MAKEP satisfies the first security requirement.

Proof. After mutually authenticating, both participants have agreed on the random numbers r_{UK} and r_{SK} by Theorem 1 and Theorem 2. Therefore, $\sigma = \sigma_{US} = r_{UK} \oplus r_{SK} = \sigma_{SU}$. □

Lemma 4. Protocol ES-MAKEP satisfies the second security requirement.

Proof. User U generates the random number r_{UK} and constructs the ciphertext $C1_{r_{UK}} = \varepsilon_{PK_S}(r_{UK})$. The encryption functions $\varepsilon_{PK}()$ and $E_K()$ are secure. Thus, only U and S know the quantity of r_{UK}. Therefore, only participants U and S can calculate the session key $\sigma = \sigma_{US} = \sigma_{SU} = r_{UK} \oplus r_{SK}$. □

Lemma 5. Protocol ES-MAKEP satisfies the third security requirement.

Proof. The proof is straightforward, since both the numbers r_{UK} and r_{SK} are randomly selected from the set $\{0, 1\}^l$. □

5. Conclusions

The paper has proposed a secure and efficient key exchange protocol. The proposed protocol outperforms the previous protocols L-MAKEP and I-MAKEP in the message round, message size, server's computations and client's online computation. These advantages make it most suitable for the environment of low power mobile communications. However, all the protocols ES-MAKEP, L-MAKEP, and I-MAKEP do not have the property of perfect forward secrecy. This strength of security requires that even the long term secret key is compromised, the session keys should remain to be safe. As can be seen from the first step in Section 2, the random number r_{UK} and session key σ_{SU} is computed if server's secret key is revealed. This suggests further research in providing a key exchange protocol with the perfect forward secrecy and all advantages of ES-MAKEP.

References

1. D. S. Wong and A. H. Chan: "Mutual authentication and key exchange for low power wireless communications," *Military Communications Conference, 2001. MILCOM 2001. Communications for Network-Centric Operations: Creating the Information Force*, IEEE, Vol. 1, 2001, pp. 39-43
2. K. Shim, "Cryptanalysis of mutual authentication and key exchange for low power wireless communications," *IEEE Communications Letters*, Vol. 7, No. 5, pp. 248-250, 2003.

3. S. Blake-Wilson, D. Johnson, and A. Menezes, “Unknown key-share attacks on the station-to-station (STS) protocols”, *The Second International Workshop on Practice and Theory in Public Key Cryptography--PKC’99*, LNCS 1560, pp. 154-170, 1999.
4. W. Diffie, P. van Oorschot, and M. Wiener, “Authentication and authenticated key exchanges,” *Designs, Codes and Cryptography*, Vol. 2, pp. 107-125, 1992.
5. J. Baek and K. Kim, “Remarks on the unknown key share attacks,” *IEICE Transactions on Fundamentals*, Vol. E83-A, No. 12, pp. 2766-2769.
6. B. S. Kaliski, Jr., “An unknown key-share attack on the MQV key agreement procotol,” *ACM Transactions on Information and System Security*, Vol. 4, No. 3, pp. 275-288, 2001.
7. J. K. Jan and Y. H. Chen, “A new efficient MAKEP for wireless communications,” *In Proceedings of the 18th International Conference on Advanced Information Networking and Application (AINA’04)*, IEEE, Volume 2, pp. 347-350, 2004.
8. A. Shamir and Y. Tauman, “Improved online/offline signature schemes,” *Advances in Cryptology-CRYPTO’01*, LNCS 2139, pp. 355-367, 2001.
9. C. Rackoff, and D. Simon, “Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack,” *Advances in Cryptology- CRYPTO’91*, LNCS 576, pp. 433-444, 1991.
10. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway, “Relations among notions of security for public key encryption schemes,” *Advances in Cryptology-CRYPTO’98*, LNCS 1462, pp. 26-46, 1998.
11. D. Dolev, C. Dwork, and M. Naor, “Non-malleable cryptography,” *Proc. of the twenty third annual ACM symposium on theory of computing*, ACM press, pp. 542-552, 1991.
12. A. Lenstra and E. Verheul, “Selecting cryptographic key sizes”, The Third International Workshop on Practice and Theory in Public Key Cryptography (PKC2000), LNCS 1751, pp. 446-465, 2000.
13. A. J. Menezes, P. C. van Oorschot and S. A. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

14. G. Miller, “Riemann’s Hypothesis and tests for primality,” *Journal of Computer and System Sciences*, ACM, Vol. 13, pp. 300-317, 1976.

15. M. Bellare and P. Rogaway, “Entity authentication and key distribution,” *Advances in Cryptology- CRYPTO’93*, LNCS 773, pp. 232-249, 1993.

16. R. Canetti and H. Krawczyk, “Analysis of key-exchange protocols and their use for building secure channels,” *Advances in Cryptology- EUROCRYPT’01*, LNCS 2045, pp. 453-474, 2001.