ISSN 1392 – 124X INFORMATION TECHNOLOGY AND CONTROL, 2010, Vol. 39, No. 2

A PARALLEL PASSWORD-AUTHENTICATED KEY EXCHANGE PROTOCOL FOR WIRELESS ENVIRONMENTS

Jung-Wen Lo

National Chung Hsing University, Department of Computer Science and Engineering
250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C.
National Taichung Institute of Technology, Department of Information Management
129 Sec. 3, San-min Rd., Taichung, Taiwan 413, R.O.C.

Shu-Chen Lin

Chaoyang University of Technology, Department of Information Management
168 Gifeng E. Rd., Wufeng, Taichung County Taiwan 413, R.O.C.

Min-Shiang Hwang

National Chung Hsing University, Department of Management Information Systems
250 Kuo Kuang Road, Taichung, Taiwan 402, R.O.C.
e-mail: mshwang@nchu.edu.tw

Abstract. Keeping the communication security between two parties and reducing the computations are important issues, especially in a wireless network environment. Based on the most suitable protocol for wireless communication, we propose a novel parallel key exchange protocol to reduce the waiting time in this paper.

Keywords: authenticated key exchange, elliptic curve discrete logarithm, key agreement, password authentication, wireless network.

1. Introduction

The Internet is widely used in the world and has influenced people's daily lives tremendously. It is a kind of open environments, so the secret information has the possibility to be wiretapped by the malevolent third party. Therefore, to ensure the information securely transmitted over the Internet is an important issue. Cryptography technique is adopted to protect the data integrity and to authenticate the validity of the identity. However, the security guarantee is more difficult in the wireless environments.

The Diffie-Hellman key agreement [1] was proposed for sharing the common session key between two parties in 1976. However, it cannot withstand the man-in-the-middle attacks, because it does not provide the mutual authentication [2–4]. To enhance Diffie-Hellman key agreement, the password-authenticated key exchange (PAKE) schemes were proposed [5–11]. Two parties pre-share a weak password (human-memorable password) to authenticate each other and then generate a session key to protect their communication over a public network.

The off-line guessing attacks in the password-authenticated key agreement protocol were considered by Bellovin and Merritt in 1992 [6]. Their protocol combines with the public-key cryptography and the secret-key cryptography that permits two parties to exchange the secret information over an insecure network by a sharing password. In 2001, Goldreich and Lindell implemented their protocol by using many techniques, such as multi-party computation, secure polynomial evaluations, and the zero-knowledge proof [7]. Hence, their protocol is too inefficient to use in the realistic world. Later, Katz et al. proposed a more efficient protocol based on the Goldreich-Lindell's protocol, but it still consumes large communication costs and computation costs [9]. Kobara and Imai proposed a pretty-simple PAKE protocol in 2002 [12]. The two communication parties use the pre-shared password to authenticate each other. They claimed that their protocol is efficient in terms of communications and computations. However, when their protocol is applied to the wireless environment, it is limited due to the restriction of storage and battery power of the mobile devices.

In a wireless network environment, the resources are limited, such as processing power, storage space, and electrical power. The elliptic curve cryptography (ECC) uses a smaller key length and less computation to achieve the same security level as RSA public-key cryptography. For example, the 160 bits key-length of ECC has almost the same level of security as the 1024 bits key-length of RSA [13, 14]. Therefore, the ECC is very suitable for using in wireless device, such as the cellular phones and PDAs [15, 16]. In 2004, Chang et al. improved Kobara-Imai's protocol for the wireless network environment [17]. Their protocol is based on the elliptic curve discrete logarithm problem (ECDLP) and has the benefit of the key block size, speed and security. In 2005, Sui et al. proposed an authenticated key agreement protocol which has a perfect forward secrecy solution for wireless mobile communication [18]. Lu et al. found that Sui et al.'s protocol had an off-line password guessing attack problem, so they proposed an improved authenticated key agreement protocol in 2007.

Due to the evolution of computer technology, we propose a parallel password-authenticated key exchange (PPAKE) protocol based on Chang et al.'s protocol to reduce the waiting time for synchronous environments. The remainder of our paper is organized as follows. In the next section, we introduce the Chang et al.'s protocols. In Section 3, the proposed parallel protocol is introduced. We have the discussion and security analysis in Section 4. Finally, we give a brief conclusion in Section 5.

2. Review of Chang et al.'s Protocol

In the wireless environment, the ability of computation and battery power in mobile devices is limited. More computations and transmissions will result in great consumption on the battery power of the client. Therefore, it is an important issue to reduce the quantity of computation and transmitting data for a client. The Chang et al.'s protocol is pretty suitable for the wireless environments [17].

The symbols used in this paper are listed as follows.

- **E**: An elliptic curve defined over a finite field $GF(q)$, where q is a prime power;
- **n**: A large prime;
- P_1, P_2: Two base points in the elliptic curve with large order n;
- $h(\cdot)$: A secure one-way hash function;
- **PW**: A password shared between the client and the server;
- $||$: The bits concatenation.

When the client wants to exchange the secret information with the server, both of them encrypt the plaintext by the session key they generated. The password PW is used to authenticate each other because only they themselves know the shared password. The proposed protocol is divided into the key agreement phase and the key confirmation phase. The protocol between the client and server is described as follows and showed in Figure 1.

Key Agreement Phase:

A1. The client chooses a random integer $d_c \in [1, n-1]$, and sends Q_c to the server, where $Q_c = d_c \cdot P_1 + PW \cdot P_2$.

A2. After receiving the registered requirement Q_c from a client, the server chooses a random integer $d_s \in [1, n-1]$ and computes $Q_s = d_s \cdot P_1$ as well as $Q_c' = Q_c - PW \cdot P_2$. Finally, the server sends $Q_s \parallel h(Key_s, Q_c')$ to the client where the key $Key_s = d_s \cdot Q_c'$.

A3. The client obtains the key Key_c by computing $Key_c = d_c \cdot Q_s$. Then, he verifies Equation (1). If it holds, the client will continue to execute the next phase, the key confirmation phase. Otherwise the client will reject the shared key Key_c.

$$h(Key_c, d_c \cdot P_1) \overset{?}{=} h(Key_s, Q_c'). \qquad (1)$$

Key Confirmation Phase:

B1. The client replies $h(Key_c, Q_s)$ to the server.

B2. Finally, the server verifies Equation (2). If it holds, the server will accept the key, otherwise the server will reject it.

$$h(Key_s, d_s \cdot P_1) \overset{?}{=} h(Key_c, Q_s). \qquad (2)$$

After passing through the key agreement phase, the session key is made by $h(d_c \cdot d_s \cdot P_1)$ and it is proved as follows:

$$\begin{aligned} Session\ Key &= h(Key_c) \\ &= d_c \cdot Q_s \\ &= d_c \cdot d_s \cdot P_1 \\ &= d_s \cdot d_c \cdot P_1 \\ &= h(Key_s). \end{aligned}$$

3. The Proposed Protocol

The PPAKE protocol is perfect for the parallel processing environment. Because the technology of the devices is rapidly improved, the capability of computation is almost equivalent between the client and server. When the client and the server have the same power or both can respond to the requests in the same time, the PPAKE protocol can speed up the key generation. It is described as follows and shown in Figure 2.

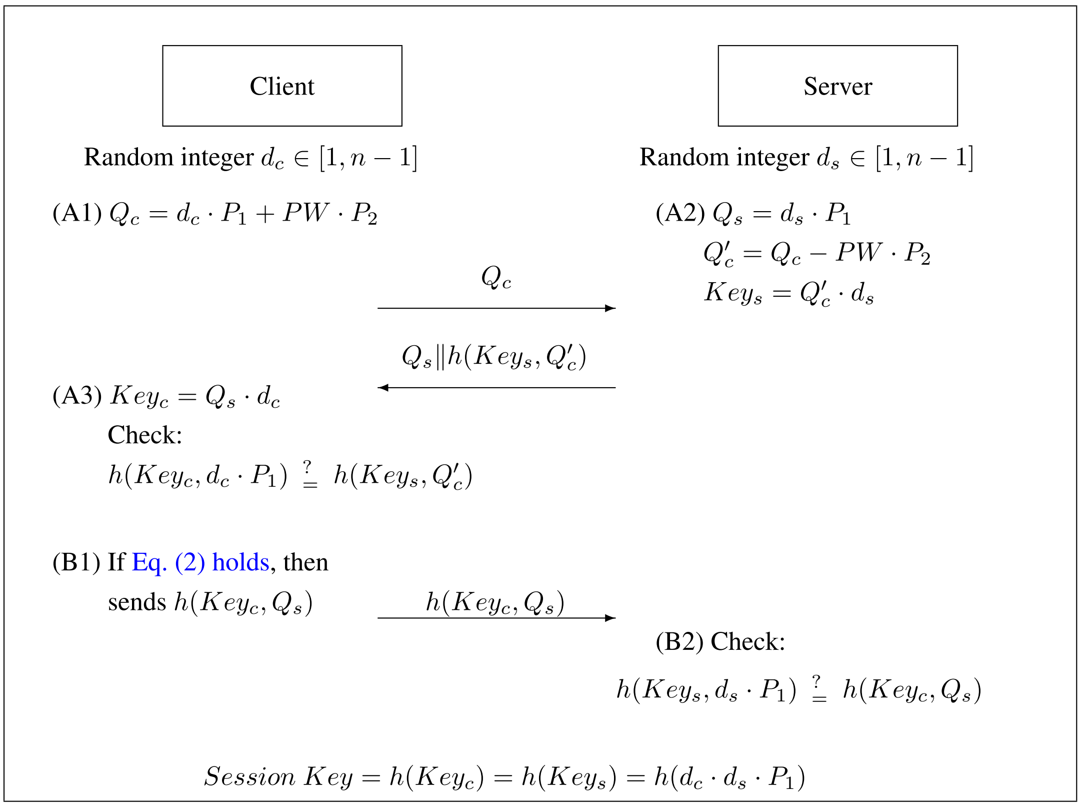
Client
Server
Random integer $d_c \in [1, n-1]$
Random integer $d_s \in [1, n-1]$
(A1) $Q_c = d_c \cdot P_1 + PW \cdot P_2$
(A2) $Q_s = d_s \cdot P_1$
$Q'_c = Q_c - PW \cdot P_2$
$Key_s = Q'_c \cdot d_s$
Q_c
$Q_s \| h(Key_s, Q'_c)$
(A3) $Key_c = Q_s \cdot d_c$
Check:
$h(Key_c, d_c \cdot P_1) \overset{?}{=} h(Key_s, Q'_c)$
(B1) If Eq. (2) holds, then
sends $h(Key_c, Q_s)$
$h(Key_c, Q_s)$
(B2) Check:
$h(Key_s, d_s \cdot P_1) \overset{?}{=} h(Key_c, Q_s)$
$Session\ Key = h(Key_c) = h(Key_s) = h(d_c \cdot d_s \cdot P_1)$

Figure 1. The password-authenticated key exchange protocol between client and server

3.1. Initial Setup

The two communication parties share a password PW before the protocol runs. Both of them compute the integer PW^{-1} before using it later, where the PW^{-1} is calculated in the field Z_n and n is a prime.

3.2. Key Agreement Phase

A1. The client chooses a random integer $d_c \in [1, n-1]$, computes $Q_c = d_c \cdot P_1 + PW \cdot P_2$, and sends Q_c to the server.

A1'. In the meantime, the server chooses a random integer $d_s \in [1, n-1]$, computes $Q_s = d_s \cdot P_1 + PW^{-1} \cdot P_2$, and sends Q_s to the client.

A2. After receiving Q_s, the client computes $Q_s{}' = Q_s - PW^{-1} \cdot P_2$ and obtains the Key_c by computing $Key_c = d_c \cdot Q_s{}'$.

A2'. After receiving Q_c, the server computes $Q_c{}' = Q_c - PW \cdot P_2$ to obtain the $Key_s = d_s \cdot Q_c{}'$.

3.3. Key Confirmation Phase

B1. The client sends $V_c = h(Key_c, Q_s{}')$ to the server. The V_c is a hash value and composed of Key_c and $Q_s{}'$.

B1'. The server sends $V_s = h(Key_s, Q_c{}')$ to the server. The V_s is a hash value and composed of Key_s and $Q_c{}'$.

B2. The client verifies Equation (3). If it holds, the client can get the session key $SK_c = h(Key_c)$, otherwise it will be rejected.

$$V_s \overset{?}{=} h(Key_c, d_c \cdot P_1). \tag{3}$$

B2'. The server checks Equation (4). If it holds, the server can get the session key $SK_s = h(Key_s)$, otherwise it will be rejected.

$$V_c \overset{?}{=} h(Key_s, d_s \cdot P_1). \tag{4}$$

In a common password-authenticated key agreement protocol, a message will be sent out during a communication, and it will be transmitted to the next stage if it receives the response message. It is the same as the stop-and-go protocol [19]. These kinds of protocols result in the low throughput. In our key agreement phase, Steps $A1$ and $A1'$ can be executed at the same time, and Steps $A2$ and $A2'$ can be executed simultaneously as well. In the key confirmation phase,

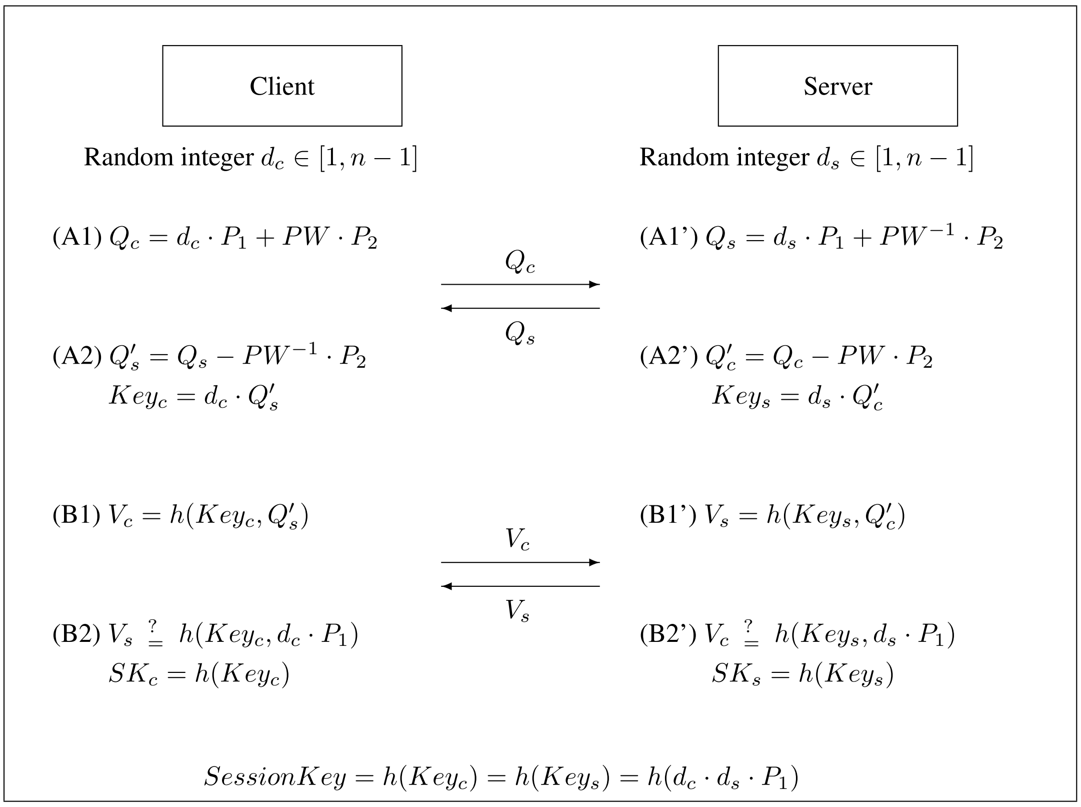
Client
Server
Random integer $d_c \in [1, n-1]$
Random integer $d_s \in [1, n-1]$
(A1) $Q_c = d_c \cdot P_1 + PW \cdot P_2$
(A1') $Q_s = d_s \cdot P_1 + PW^{-1} \cdot P_2$
Q_c
Q_s
(A2) $Q'_s = Q_s - PW^{-1} \cdot P_2$
$Key_c = d_c \cdot Q'_s$
(A2') $Q'_c = Q_c - PW \cdot P_2$
$Key_s = d_s \cdot Q'_c$
(B1) $V_c = h(Key_c, Q'_s)$
(B1') $V_s = h(Key_s, Q'_c)$
V_c
V_s
(B2) $V_s \overset{?}{=} h(Key_c, d_c \cdot P_1)$
$SK_c = h(Key_c)$
(B2') $V_c \overset{?}{=} h(Key_s, d_s \cdot P_1)$
$SK_s = h(Key_s)$
$SessionKey = h(Key_c) = h(Key_s) = h(d_c \cdot d_s \cdot P_1)$

Figure 2. The password-authenticated key exchange protocol in parallel processing environment

Steps $B1$ and $B1'$ can be executed at the same time, and Steps $B2$ and $B2'$ can be executed simultaneously as well. Evidently, our protocol speeds up the session key generation by using the parallel processing mechanism without the waiting time.

In addition, we can make some minor modifications to increase the performance. If the client and the server store the value of $PW \cdot P_2$ and $PW^{-1} \cdot P_2$ in their storage, the number of the elliptic curve multiplication can be reduced to two times. Considering the light-weight client, the values d_c and Q_c can be pre-determined and stored in advance to improve the efficiency. Therefore, the client requires the less time-consuming.

4. Discussions and Security Analysis

In this section, the requirements and attacks analyses of the proposed protocols are presented.

4.1. Requirements of the Key Agreement Protocol

The key agreement protocol should satisfy the following requirements. We show that our protocol meets the requirements.

1. **Session Key Security.**
Only the communication client and the server have the same session key. In our protocol, the session key is generated by the random numbers chosen by both parities individually, and the random numbers are also protected by the ECDLP and the common password. When an attacker guesses a password pw' to obtain a value $Q''_c = Q_c - pw' \cdot P_2$, it is hard to obtain the d_c from the point Q''_c due to the ECDLP. Also, it is very difficult to derive the d_s from the Q_s. Therefore, the session key is only known by the communication client and the server.

2. **Mutual Authentication.**
The client and the server can authenticate each other to make sure that they are communicating with the right party. When the protocol runs at Step B2, the client can authenticate the server by Equation (3) because only the correct server has the shared PW for computing the message Q_s and V_s. Also, the server authenticates the client by Equation (4). Therefore, the protocol has the property of the mutual authentication.

3. **Perfect Forward/Backward Secrecy.**
When a session key is compromised, the attacker cannot derive any previous/following session key. The integer d_c chosen by the client and d_s chosen by the server are randomly generated in every session, so

we can guarantee that the session key is unique in every session. All session keys are not related, so the PPAKE protocol does not have any perfect forward/backward security problem.

4. **Known-key Security.**
The known-key security should guarantee that the session key produced by every session is unique. The client chooses random number d_c and the server chooses random number d_s individually in every session to have a distinguishable session key $h(d_c \cdot d_s \cdot P_1)$, so each session has a unique session key.

5. **Key Control Security.**
For security reason, the session key cannot be chosen by client or server [20]. In our protocol, the session key, $h(Key_c) = h(Key_s) = h(d_c \cdot d_s \cdot P_1)$, is computed by the number d_c chosen by the client and d_s chosen by the server. Therefore, neither one can determinate the session key alone. The session key is determined by both communication parties.

4.2. Attacks Analyses

We show that the proposed protocol can resist the following four attacks.

1. **Man-in-the-middle Attack.**
The Man-in-the-middle Attack is one in which an attacker makes independent connections with the client and server, and controls the entire conversation by relaying messages between them. In our protocol, the attacker cannot know the pre-shared password, so he only can blindly forge a message. Without the password, the attacker cannot compute Q'_c or Q'_s and derive the Key_s or Key_c. Therefore, he only can give a mess message instead of the correct V_c or V_s. Of course, the mess message will be detected at Step $B2$ or $B2'$, so the man-in-the-middle attack cannot succeed.

2. **Password Guessing Attack.**
The password guessing attack is that an adversary guesses the password on-line or off-line. It is very easy to detect the on-line password guessing attack by counting the number of guessing. But the off-line guessing attack is more difficult to prevent. The adversary intercepts $\{Q_c = d_c \cdot P_1 + PW \cdot P_2\}$, $\{Q_s = d_s \cdot P_1 + PW^{-1} \cdot P_2\}$, $\{V_c = h(Key_c, Q'_s)\}$ and $\{V_s = h(Key_s, Q'_c)\}$ in the communication. The adversary guesses a password PW' and computes $Q'_c = Q_c - PW' \cdot P_2$. However, he cannot derive the d_s or d_c due to the elliptic curve discrete logarithm problem (ECDLP), so he cannot have the correct Key_s or Key_c to compare the message V_c or V_s. Therefore, it is difficult to guess the password off-line by an adversary.

3. **Impersonation Attack.**
The impersonation attack is that an adversary impersonates one of the communication parties in order to use the service he wishes. In our protocol, the adversary can successfully impersonate any one communication party only if he has the pre-shared password. However, it is difficult to obtain the password from the intercepted message over Internet. If the adversary can obtain the flaw messages, he should face to solve ECDLP if he wants to compute the PW from Q_c. Without the PW, he cannot compute the Key_s to pass through the key confirmation phase. Besides, the Key_s is also protected by the one-way hash function when the adversary has the message V_s. Therefore, the impersonation attack can be prevented in PPAKE protocol.

4. **Reflection Attack.**
The reflection attack is a serious problem in the parallel processing mechanism. The attack indicates that when an adversary re-sends the message generated by one party and may pass through the verification phase. If the attacker intercepts the message $Q_c = d_c \cdot P_1 + PW \cdot P_2$ from the legal client and re-send Q_c instead of Q_s and V_c instead of V_s to the client, the client will reject the connection in the key confirmation phase. The way that the client finds out the confliction is stated as follows:

$$
\begin{aligned}
(A2)\ Q'_s &= Q_c - PW^{-1} \cdot P_2 \\
&= (d_c \cdot P_1 + PW \cdot P_2) - PW^{-1} \cdot P_2. \\
Key_c &= d_c \cdot Q'_s \\
&= d_c \cdot (Q_c - PW^{-1} \cdot P_2) \\
&= d_c \cdot (d_c \cdot P_1 + PW \cdot P_2 - PW^{-1} \cdot P_2). \\
(B1)\ V_c &= h(Key_c, Q'_s). \\
(B2)\ V_s &= h(Key_c, Q'_s) \\
&= h(Key_c, (d_c \cdot P_1 + PW \cdot P_2 - PW^{-1} \cdot P_2)) \\
&\neq h(Key_c, d_c \cdot P_1).
\end{aligned}
$$

Therefore, our protocol can resist the reflection attack.

5. Conclusions

The PPAKE protocol is introduced in this article. It is a parallel protocol for generating a session key between two parties and is less time-consuming by parallel processing. We showed that it reaches the requirements of key agreement protocol and is secure enough to be used.

If the client and the server store the values of $PW \cdot P_2$ and $PW^{-1} \cdot P_2$ in the storage, the executing time can be reduced. So, the protocol can be applied in the light-weight wireless client. Furthermore, if the server stores the values too, it also can save the time and computing power and then supply more services.

Therefore, the PPAKE protocol is a good choice of key agreement protocol in the wireless environments or Internet.

Acknowledgement

This work was supported in part by the Taiwan Information Security Center (TWISC) and the National Science Council under the grants NSC95-2218-E-001-001, and NSC95-2218-E-011-015.

References

[1] **W. Diffie, M. E. Hellman.** New directions in cryptography. *IEEE Transactions on Information Theory*, 1976, IT-22 (6), 644 - 654.

[2] **M.-S. Hwang, J.-W. Lo, C.-H. Liu.** Enhanced of key agreement protocols resistant to a denial-of-service attack. *Fundamenta Informaticae*, 2004, 61 (3-4), 389 - 398.

[3] **E. J.-L. Lu, M.-S. Hwang.** An improvement of a simple authenticated key agreement algorithm. *Pakistan Journal of Applied Sciences*, 2002, 2 (1), 64 - 65.

[4] **E. J.-L. Lu, C.-C. Lee, M.-S. Hwang.** Cryptanalysis of some authenticated key agreement protocols. *International Journal of Computational and Numerical Analysis and Applications*, 2003, 3 (2), 151 - 157.

[5] **M. Bellare, D. Pointcheval, P. Rogaway.** Authenticated key exchange secure against dictionary attacks. *in: Advances in Cryptology - EUROCRYPT'00*, 2000, 139 - 155.

[6] **S. M. Bellovin, M. Merritt.** Encrypted key exchange: Password-based protocols secure against dictionary attacks. *in: Proceedings of 1992 IEEE Computer Society Conference on Research in Security and Privacy*, 1992, 72 - 84.

[7] **O. Goldreich, Y. Lindell.** Session-key generation using human passwords only, *in: Advances in Cryptology, CRYPTO'01, Lecture Notes in Computer Science*, 2001, 2139, 408 - 432.

[8] **W.-S. Juang, J.-L. Wu.** Efficient user authentication and key agreement with user privacy protection. *International Journal of Network Security*, 2008, 7 (1), 120 - 129.

[9] **J. Katz, R. Ostrovsky, M. Yung.** Efficient password-authenticated key exchange using human-memorable passwords, *in: Proceedings of EUROCRYPT'2001, Austria, Lecture Notes in Computer Science*, 2001, 2045, 475 - 494.

[10] **C. C. Lee, T. C. Lin, M. S. Hwang.** A key agreement scheme for satellite communications. *Information Technology And Control*, 2010, 39 (1), 43 - 47.

[11] **J. L. Tsai.** Weaknesses and improvement of Hsu-Chuang's user identification scheme. *Information Technology And Control*, 2010, 39 (1), 48 - 50.

[12] **K. Kobara, H. Imai.** Pretty-simple password-authenticated key-exchange protocol proven to be secure in the standard model. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 2002, E85-A (10), 2229 - 2237.

[13] **T. C. Lin.** Algorithms on elliptic curves over fields of characteristic two with non-adjacent forms. *International Journal of Network Security*, 2009, 9 (2), 117 - 120.

[14] **D. Yong, Y. F. Hong, W. T. Wang, Y. Y. Zhou, and X. Y. Zhao.** Speeding scalar multiplication of elliptic curve over $GF(2^{mn})$. *International Journal of Network Security*, 2010, 11 (2), 70 - 77.

[15] **S. A. Vanstone.** Next generation security for wireless: elliptic curve cryptography. *Computers and Security*, 2003, 22 (5), 412 - 415.

[16] **C.-C. Yang, T.-Y. Chang, M.-S. Hwang.** A new anonymous conference key distribution system based on the elliptic curve discrete logarithm problem. *Computer Standards & Interfaces*, 2003, 25 (2), 141 - 145.

[17] **T.-Y. Chang, C.-C. Yang, C.-M. Chen.** Improvement on pretty-simple password authenticated key-exchange protocol for wireless networks. *Informatica*, 2004, 15 (2), 161 - 170.

[18] **A.-F. Sui, L. C. Hui, S. Yiu, K. Chow, W. Tsang, C. Chong, K. Pun, H. Chan.** An improved authenticated key agreement protocol with perfect forward secrecy for wireless mobile communication. *IEEE Wireless Communications and Networking Conference*, 2005, 4, 2088 - 2093.

[19] **D. E. Comer.** Computer networks and Internets, 5th Edition. *Pearson Prentice Hall*, 2008.

[20] **C. Mitchell, M. Ward, P. Wilson.** Key control in key agreement protocols. *Electronics Letters*, 1998, 34 (10), 980 - 981.

Received December 2009.