情報保護學會論文誌
第 22 卷 第 3 號, 2012. 6

새로운 C2C-PAKA 프로토콜의 안전성 연구

변 진 욱†‡
평택대학교 정보통신학과

On the Security of a New C2C-PAKA Protocol

Jin Wook Byun†‡
Pyeongtaek University, Department of Information and Communication

요 약

단말 간 사용자의 안전성 확보를 위해, 동일한 패스워드를 가정하여 두 사용자를 인증하는 환경은 실용적이지 않다. 왜냐하면, 사용자들은 각자 고유의 다른 패스워드를 암기하고 있기 때문이다. 이를 해결하기 위해 서로 다른 패스워드를 이용한 단말간의 키 교환 (EC2C-PAKA) 프로토콜이 서로 다른 영역의 환경 (cross-realm setting) 에서 제안되었다. 최근에, 이러한 EC2C-PAKA 프로토콜에 대한 취약점이 Feng과 Xu에 의해서 주장되었다. 그들은 EC2C-PAKA 프로토콜이 패스워드 가장공격에 취약함을 주장하였다. 그들은 또한 패스워드 가장공격에 강인한 프로토콜을 제안했다. 본 논문에서는 Feng과 Xu가 제안한 공격이 옳지 않음과 EC2C-PAKA 프로토콜이 여전히 패스워드 가장 공격에 강인함을 보인다. 반대로, Feng과 Xu가 향상 시킨 프로토콜이 A 영역에서 *Alice*의 패스워드를 알고 있는 서버가 B 영역에 있는 *Bob*을 가장할 수 있는 가장 공격에 취약함을 보인다. 이에 대한 대처방안도 논의한다.

ABSTRACT

To achieve an entire end-to-end security, the classical authentication setting such that all participants have a same password is not practical since a password is not a common secret but a personal secret depending on an individual. Thus, an efficient client to client different password-based authenticated key agreement protocol (for short, EC2C-PAKA) has been suggested in the cross-realm setting. Very recently, however, a security weakness of the EC2C-PAKA protocol has been analyzed by Feng and Xu. They have claimed that the EC2C-PAKA protocol is insecure against a password impersonation attack. They also have presented an improved version of the EC2C-PAKA protocol. In this paper, we demonstrate that their claim on the insecurity of EC2C-PAKA protocol against a password impersonation attack is not valid. We show that the EC2C-PAKA protocol is still secure against the password impersonation attack. In addition, ironically, we show that the improved protocol by Feng and Xu is insecure against an impersonation attack such that a server holding password of *Alice* in realm *A* can impersonate *Bob* in realm B. We also discuss a countermeasure to prevent the attack.

Keywords: Password authentication, Key exchange, Different Password, Security Analysis

I. Introduction

A human memorable password has steadily been a popular mean for authenticating clients over the Internet. The reason is that the password has strengths such that it is

접수일(2011년 10월 5일), 수정일(2011년 12월 5일), 게재확정일(2011년 12월 26일)
† 주저자, jwbyun@ptu.ac.kr
‡ 교신저자, jwbyun@ptu.ac.kr

easy to be memorized and implemented. Such advantage not only brings clients much convenience but also provides system administrators with economic profits when implementing an authentication system in practice. In fact, most authentication systems rely on password authentication to verify the identity of a user before allowing user to login and obtain various network resources. In addition to the authentication, securely agreeing a common session key between a client and a server is one of the indispensable services for a secure communication over the Internet. The agreed session keys are used to guarantee confidentiality by encrypting (or decrypting) confidential messages and verifying message authentication codes.

In order to provide both the authentication and the confidentiality, an efficient and secure integration of a password-based authentication and a key agreement protocol has been widely studied in the literature. Generally, we call this compound notion as PAKA (password-based authenticated key agreement). However, it has been a challengeable task to design the PAKA protocol satisfying both security and efficiency [4,5,6,7,8,9,10,11,12,13,14,15,17]. It is mainly due to that a selected password from a small space allows an adversary to mount off-line dictionary attacks in which the adversary tries all possible combinations of secret values such as telephone number and identification number in a given small set of dictionary. Nevertheless, many secure PAKA protocols have been suggested and studied in terms of general constructions using minimum cryptographic primitives and how to securely extend to multi-party setting.

However, most PAKA protocols have concentrated on a classical authentication between a client and a server. The issue is that the PAKA protocol itself has limitations to meet various requests of authentication in an end-to-end situation between realms where a client *Alice* in a realm *A* wants to establish a secure session with a client *Bob* in a realm *B*. To achieve end-to-end security, the setting such that all participants have a same password is not practical since a password is not a common secret but a secret depending on an individual. Thus, following question is naturally raised: *If the password is already pre-distributed in a secure manner, respectively, why don't we generate a common session key by using different passwords?*

1.1 Related Works and Contributions

To address the above practical issue, Byun et al. have first designed a client to client password authenticated key agreement (C2C-PAKA) with different password which enables two clients only holding own password to mutually authenticate and derive a common session key. Since then, the C2C-PAKA protocol has been extensively analyzed and revised under the various security aspects such as kinds of impersonation and known key attacks based on the diverse attacker's behaviors [16]. In fact, a few improved protocols have been suggested and subsequently have been found to be flawed. Despite of many attempts of cryptanalysis, there are still possibilities that new security breaches can be found because their all security analysis are based on the heuristic approach. In 2007, Byun et al. have first attempted to establish a formal security model for C2C-PAKA and presented a new efficient C2C-PAKA (EC2C-PAKA) with formal security proof [2].

However, in 2009, Feng and Xu have

promptly pointed out that the EC2C-PAKA protocol is not secure in terms of a password impersonation attack. Generally, the security on password impersonation means that revealment of client *Alice*′s password should not enable an outside attacker to share a session key with *Alice* by masquerading as any other client, *Bob* [1]. They insisted that an attacker A holding a password of *Alice* is able to make forged authentication messages to be able to pass a verification phase by *Alice* without being noticed by *Alice*. Thus, A can share a session key with *Alice* by masquerading as *Bob*.

In this paper, first of all, we show that the EC2C-PAKA protocol is secure against a password impersonation attack. Concretely, we demonstrate that the original EC2C-PAKA protocol does not allow the attacker A′ even obtaining *Alice*′s password to pass a verification phase for *Alice*. Second, we examine the security of the improved protocol which has been suggested as a countermeasure against the password impersonation attack by Feng and Xu [1]. Interestingly, the protocol is found to be susceptible against an impersonation attack in which a malicious server in realm *A* can impersonate any client in realm *B*. We show how it is possible in the protocol. We finally discuss a countermeasure against the attack.

1.2 Organization

Next chapter we revisit an EC2C-PAKA protocol by Byun et al.[2] In Chapter 3, we show that the EC2C-PAKA protocol is secure against a password impersonation attack. In Chapter 4, we describe the improved protocol by Feng and Xu and demonstrate that it is insecure against an impersonation attack. We conclude in Chapter 5.

II. Overview of EC2C-PAKA Protocol

We assume a large safe prime order q over Z_q^*. A hash function H is defined as $H(\cdot):\{0,1\}^* \rightarrow 0,1^l$ where l is the output size of hash function. Two encryption functions are used; one is an ideal cipher ε such as one in [3] which is a random one-to-one function such that $E_K: M \rightarrow C$ where |M| = |C| and the other function is a CCA (chosen ciphertext attack) secure symmetric encryption E. Notations throughout paper are listed in Table 1.

[Table 1] Notations

Notations	Meaning
ID_A, ID_B	identifiers of *Alice* and *Bob*
R, R'	ephemeral Diffie-Hellman keys for Alice and KDC_A, *Bob* and KDC_B
K	a common symmetric key pre-distributed for KDC_A and KDC_B
sk	a session key agreed between *Alice* and *Bob*
k	a common key distributed for *Alice* and *Bob*
Ticket_ B	a service ticket for *Bob*
L	a lifetime of $Ticket_B$
MAC_k(m)	an output of MAC applied key k for a message m
||	two adjacent messages are concatenated.
$Sign\_X(m)$	a signature of message m signed by X′s secret key.
$E\_X(m)$	an encryption of message m with X′s public key.

2.1 Protocol Preliminaries

Preliminaries for a protocol run are as follows.

1. g and q are global public parameters shared by all protocol participants, where q is a prime order and g is a

generator over a cyclic group Z_q^*.

2. *Alice* (*Bob*) shares her password *pwa* (*pwb*) with server KDC_A (KDC_B, respectively) by using algorithms G_{pw} and R.

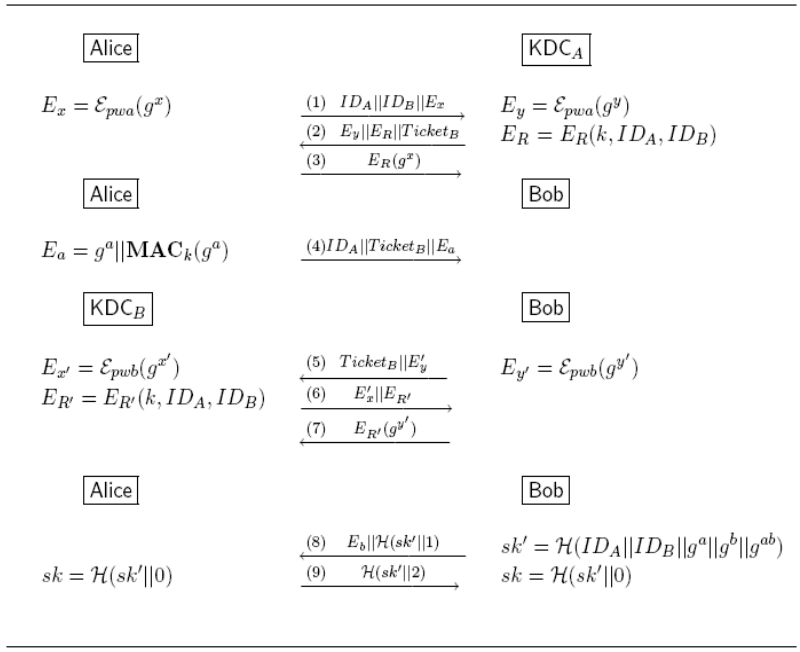
2.2 Protocol Description of EC2C-PAKA

The EC2C-PAKA protocol is illustrated in Figure 1. It works as follows.

1. *Alice* chooses a random value x from Z_q^* randomly then computes g^x and sends $E_x = \varepsilon_{pwa}(g^x)$ to KDC_A along with ID_A and ID_B.
2. KDC_A obtains g^x by decrypting E_x, chooses $y \in Z_q^*$ randomly, and computes $E_y = \varepsilon_{pwa}(g^y)$ and $R = H(g^{xy})$. KDC_A also generates a random key k from Z_q^* for *Alice* and *Bob* and computes $E_R = E_R(k, ID_A, ID_B)$. KDC_A specifies L, a lifetime of $Ticket_B$. Then KDC_A makes $Ticket_B (= E_K(k, ID_A, ID_B, L))$ and sends E_y, E_R, and $Ticket_B$ to *Alice*.
3. Upon receiving the message from KDC_A, *Alice* computes an ephemeral key R and decrypts E_R to obtain the distributed key k. *Alice* also checks whether ID_A and ID_B are correct or not. The encrypted message, $E_R(g^x)$ is also sent to *Bob* for authentication.
4. *Alice* generates a random value $a \in Z_q^*$ and makes $E_a = (g^a \| MAC_k(g^a))$. Then she forwards ID_A, E_a, and $Ticket_B$ to *Bob*.
5. *Bob* chooses $y' \in Z_q^*$ randomly and computes $E_{y'} = E_{pwb}(g^{y'})$. Then he sends $E_{y'}$ and $Ticket_B$ to KDC_B.
6. KDC_B obtains k, L, and ID_A by decrypting $Ticket_B$ by using its key K. KDC_B first examines the validity of $Ticket_B$ by checking the lifetime L and ID_A. If the validation check is successful, KDC_B selects $x' \in Z_q^*$ randomly and computes $E_{x'} = (\varepsilon_{pwb}(g^{x'}))$ and $E_{R'} (= E_{R'}(k, ID_A, ID_B)$ where R' is $H(g^{x'y'})$. KDC_B finally sends $E_{x'}$ and $E_{R'}$ to *Bob*.
7. *Bob* decrypts $E_{x'}$ and computes $R' (= H(g^{x'y'}))$. Then *Bob* decrypts $E_{R'} = E_{R'}(k, ID_A, ID_B)$ to get the key k. *Bob* computes $E_{R'}(g^{y'})$ for a random $y' \in Z_q^*$ and send it to KDC_B for authentication.
8. *Bob* decrypts $E_{x'}$ and computes R'. Then *Bob* decrypts $E_{R'}$ to get the key k. Using the key k, *Bob* checks g^a by verifying the previously received E_a. *Bob* generates a random value $b \in Z_q^*$ and makes $sk' (= H(ID_A, ID_B, g^a, g^b, g^{ab})$ and $E_b = E_b(g^b \| MAC_k'(g^b))$. Finally *Bob* sends E_b to *Alice*. Upon receiving the message E_b, *Alice* also generates a common session key sk.
9. Upon receiving the message, *Alice* confirms the authenticator by using sk' and makes $H(sk' \| 2)$. *Alice* sends this back to *Bob*. If the confirmation processes are successful, then *Alice* and *Bob* generate a common session key $sk = H(sk' \| 0)$.

III. Analysis of Attack on EC2C-PAKA Protocol

In this section, we describe an attack scenario suggested by Feng and Xu and demonstrate that the attack is invalid. First, an attacker A′ is assumed to have a password *pwa* of *Alice* and then tries to masquerade as *Bob*. Finally the goal of the attacker is to share a common session key with *Alice* as *Bob*.



〔Figure 1〕 The original EC2C-PAKA Protocol

3.1 A scenario of password impersonation attack

1. A′ captures message, E_x, ID_A, ID_B from the step (1). A′ can decrypt E_x and obtain g^x. Values, $\bar{k}, y \in Z_q^*$, $Ticket_{\bar{B}}$ are chosen randomly by A′ then $E_y = \varepsilon_{pwa}(g^y)$, $R = H(g^{xy})$, $E_{\bar{R}} = E_R(\bar{k}, ID_A, ID_B)$ are calculated by A′. A′ sends $E_y, E_{\bar{R}}$, $Ticket_{\bar{B}}$ to *Alice* in the step (2) instead of KDC_A. Then *Alice* keeps the key $\bar{k}$ chosen by A′.
2. From the step (4) of the protocol, A′ obtains g^a and selects $b \in Z_q^*$ randomly. In the step (8), A′ sends $E_{\bar{b}} = g^b||MAC_{\bar{k}}(g^b)$ to *Alice* as if it is originated from *Bob*.
3. The forged message will be valid for *Alice* during the step (8) since A′ has the same $\bar{k}$ as *Alice*. Finally, A′ can be successfully authenticated by *Alice* as a client *Bob* and also shares a session key $sk = H(ID_A||ID_B||g^a||g^b||g^{ab})$ with *Alice*.

3.2 Analysis of the Attack

The attack mainly dues to an assumption that an attacker A′ obtaining pwa can go through the verification test at the side of Alice during step (8). To be precisely, the test of step (8) verifies MAC tag with the common key $\bar{k}$ and g^b which were chosen randomly by A′, as follows.

$$E_{\bar{b}} = g^b||MAC_{\bar{k}}(g^b)$$

The above verification is always valid since Alice maintains the same key $\bar{k}$. However, before(or even after) sending the above forged tag message to Alice in the step (8), A′ must pass critical verification processes remaining in the step (6) and (8). Since A′ cannot pass the verification proc-

esses, every involving participant finally gets to recognize that all processes are invalid. To be precisely, let's go back the above attack scenario and consider the impersonation attack again.

A′ sends the followings to Alice in the step (2).

$$E_y = \varepsilon_{pwa}(g^y)$$
$$E_R = E_R(\bar{k}, ID_A, ID_B)$$
$$Ticket_{\bar{B}}$$

The point is that $\bar{k}$, y and $Ticket_{\bar{B}}$ were randomly created by A′.

1. As explained earlier, A′ can send $E_b = g^b \| MAC_{\bar{k}}(g^b)$ to Alice in the step (8) as if it is originated from Bob. However, before doing it, the following verification tests in the step (6) and (8) prevent A′ from impersonating as Bob.

$$D_K(Ticket_B) = ID_A, ID_B, L$$
$$E_a = g^a \| MAC_{\bar{k}}(g^a)$$

First, in the step (5), the ticket $Ticket_{\bar{B}}$, which is randomly selected by A′, is transferred to KDC_B. In the step (6), KDC_B decrypts it and obtains a distinct key $k'(\neq \bar{k})$ and verifies a validity check of $Ticket_{\bar{B}}$ through the above equation (2). However, the inconsistent key k' always results in failing of verification in the equation (2). It is attributed to a fact that the common key k of $Ticket_B$ is selected by the valid KDC_A and securely protected by a private key K hence nobody knows it except the valid KDC_B and KDC_A. Second, in the step (8), Bob should do check the tag message $E_a = g^a \| MAC_{\bar{k}}(g^a)$ previously received in the step (4). Originally, this tag message is calculated using the key k chosen by the valid Bob. However, the message of step (2) is forged by A′ as the form of equation (1) and it makes Alice to possess the key $\bar{k}$. Since it is not consistent with the key k, the verification of step (8) does not always succeed. Therefore, A′ never be able to generate both a valid ticket and MAC tag of E_a to satisfy the equation (2) and (3) without knowing K.

3. Nevertheless, in the step (8), A′ may try to generate a session key $sk' = H(ID_A \| ID_B \| g^a \| g^b \| g^{ab})$ with Alice by just sending $E_b = g^b \| MAC_{\bar{k}}(g^b)$ as quick as possible. However, the valid client Bob already has noticed that the verification has been invalid in the steps (5) and (8). Even if the message of (8) has been sent to Alice, Bob always still has a chance to reject Alice and be able to let all parties know the current protocol is failed.

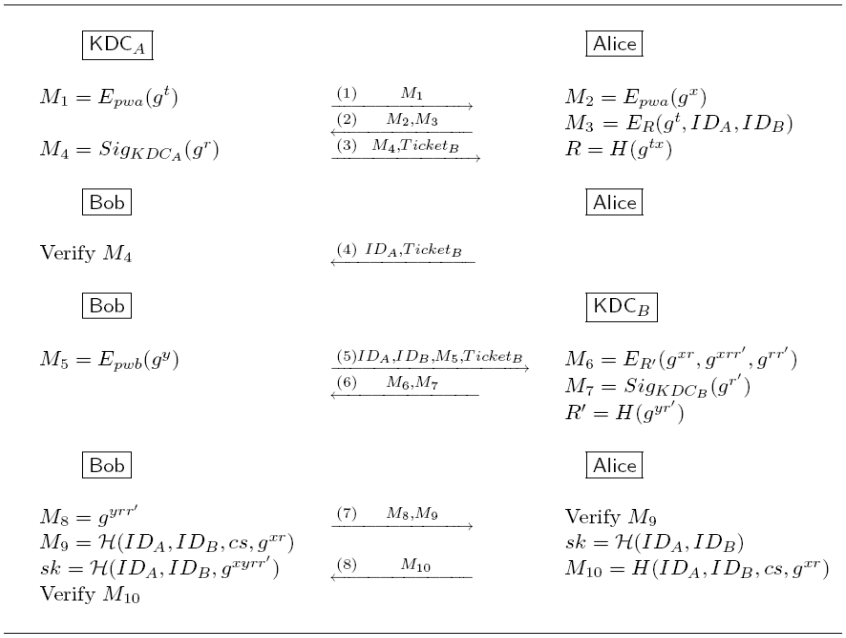
IV. Vulnerability of an improved C2C-PAKA protocol

In this section, we briefly introduce an improved version of C2C-PAKA protocol suggested by Feng and Xu [1] and demonstrate that it is insecure against an impersonation attack where a server KDC_A in the realm A is able to impersonate Bob in the realm B.

4.1 Overview of an improved C2C-PAKA protocol

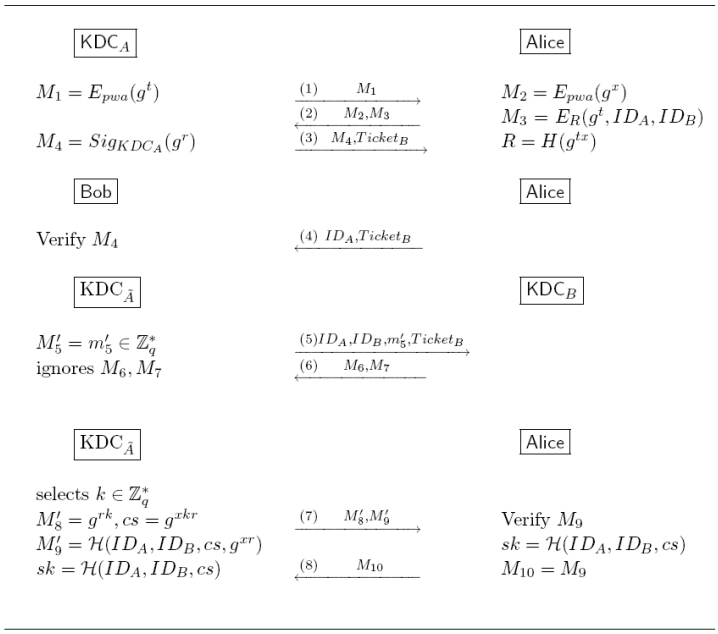
The protocol consists of (8) steps. It is illustrated in Figure 2.

1. *Alice* sends the message ID_A, ID_B to KDC_A. KDC_A randomly selects $t \in Z_q^*$ and encrypts g^t with pwa then transfer $M_1 = \varepsilon_{pwa}(g^t)$.



〔Figure 2〕 The improved EC2C-PAKA Protocol by Xu and Feng

2. *Alice* can obtain g^t and randomly chooses $x \in Z_q^*$. Then she calculates $M_2 = \varepsilon_{pwa}(g^x)$, $R = H(g^{tx})$, and $M_3 = E_R(g^t, ID_A, ID_B)$. Finally, *Alice* sends M_2, M_3 to KDC_A.
3. On receiving the message from *Alice*, KDC_A decrypts M_2 and obtains g^x and checks the validity of M_3 by using R. KDC_A randomly chooses $r \in Z_q^*$ and calculates $M_4 = Sig_{KDC_A}(g^r)$ and $Ticket_B = PE_{KDC_B}(Sig_{KDC_A}(g^r, g^{xr}, ID_A, ID_B, L))$ and sends M_4, $Ticket_B$ to *Alice*.
4. On receiving the message from S_A, *Alice* checks the signature of M_4 is valid. *Alice* computes g^{xr} and forwards ID_A, $Ticket_B$ to *Bob*.
5. *Bob* selects $y \in Z_q^*$ and calculates $M_5 = E_{pwb}(g^y)$ then the messages $ID_A, ID_B, M_5, Ticket_B$ are forwarded to KDC_B.
6. KDC_B obtains g^y from M_5. KDC_B decrypts $Ticket_B$ and verify the signature of KDC_A. KDC_B obtains g^{xr} and g^r and selects $r' \in Z_q^*$ randomly. KDC_B also computes the message $g^{yr'}, g^{xrr'}, g^{rr'}$ and encrypts it with $R'(= H(g^{yr'}))$, then makes $M_6 = E_{R'}(g^{xr}, g^{xrr'}, g^{rr'})$, $M_7 = Sig_{KDC_B}(g^{r'})$, which are sent to *Bob*.
7. With the message from KDC_B, *Bob* checks M_7 and computes $R' = H(g^{r'y})$ and computes $g^{xr}, g^{xrr'}$ by decrypting M_6. *Bob* calculates $cs = g^{xyrr'}$, $M_8 = g^{rr'y}$, and $M_9 = H(ID_B, ID_A, cs, g^{xr})$. *Bob* sends M_8, M_9 to *Alice* for a session key confirmation.
8. With the message M_8, M_9, *Alice* computes $cs = g^{xyrr'}$ and computes $H(ID_B, ID_A, cs, g^{xr})$ which is verified with the message M_9. If it holds, *Alice* authenticates *Bob*. *Alice* computes $M_{10} = H(ID_A, ID_B, cs, g^{xr})$. M_{10} is also sent to



〔Figure 3〕 A password impersonation attack on Feng and Xu's protocol

Bob for a session key confirmation. With the message M_{10}, *Bob* computes $H(ID_A, ID_B, cs, g^{xr})$ and verifies with M_{10}. If it succeeds, *Bob* authenticates *Alice*. A common session key between *Alice* and *Bob* is $sk = H(ID_A, ID_B, g^{xyrr'})$

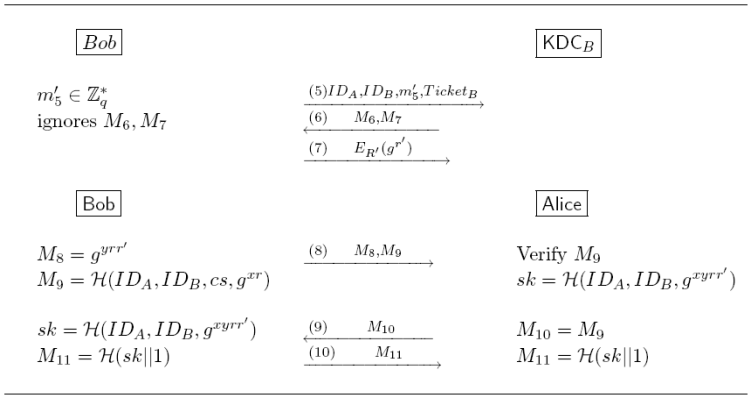
4.2 Vulnerability of the improved C2C-PAKA scheme by Xu and Feng

Xu and Feng presented a new improved version of C2C-PAKA protocol [1]. We demonstrate that the improved scheme is weak against an impersonation attack by a malicious server $KDC_{\bar{A}}$. If we assume a malicious $KDC_{\bar{A}}$ which keeps a password *pwa* for a client *Alice*, then it can impersonate a client *Bob* in realm B. The attack is performed as follows. The scenario is illustrated in Figure 3.

1. Since the malicious server $KDC_{\bar{A}}$ has a password *pwa*, $KDC_{\bar{A}}$ can decrypt M_2, M_3 from the step (2) of the protocol and then keeps g^{xr}. In order to impersonate *Bob*, $KDC_{\bar{A}}$ should send messages of step (5) instead of *Bob*. To do so, $KDC_{\bar{A}}$ randomly chooses $m_5' \in Z_q^*$ and sends $ID_A, ID_B, m_5', Ticket_B$ to *Bob*.
2. In the step (6), KDC_B sends back M_6, M_7 to $KDC_{\bar{A}}$ and then $KDC_{\bar{A}}$ simply ignores the message M_6, M_7.
3. For the step (7), $KDC_{\bar{A}}$ first chooses a random value $\phi \in Z_q^*$ and calculates $cs = (g^{xr})^{\phi}$ with the value of g^{xr} obtained from the step (2). The forged message M_8', M_9' for the step (7) are calculated as follows.

$$M_8' = g^{r\phi}$$
$$M_9' = H(ID_A, ID_B, g^{xr\phi}, g^{xr})$$

4. On the step (7), the valid client *Alice*



〔Figure 4〕 Countermeasure

computes $cs = (M_8')^x = g^{xr\phi}$ by using the chosen random value x and verifies M_9' by using the computed cs and g^{xr}. Hence no fail happens.

5. In the step (8), $KDC_{\bar{A}}$ receives M_{10} and generates $sk = H(ID_A, ID_B, g^{xr\phi})$. Therefore, the malicious $KDC_{\bar{A}}$ succeeds in impersonating *Bob* and generating a session key sk.

4.3 Countermeasure

In order to deter the impersonation attack from the malicious KDC_A, we need a kind of verification check which enables KDC_B to recognize itself that the protocol is being cheated, and finally it is able to let all involving participants know the current protocol fails. Indeed, the difference between two protocols [1, 2] in terms of structure of communication is the absence of the authentication from KDC_B to *Bob*, which actually induces an impersonation attack by an insider server KDC_A. Basic idea to deal the attack is to simply inject authentications into weakness point of the protocol. Concretely, two tags for authentications are required. The first authentication tag is added after the step (6), as a form of $E_{R'}(g^{r'})$, which is transferred from *Bob* to KDC_B. The second authentication tag is for *Alice* to authenticate *Bob*, as a form of hash based authenticator $H(sk||1)$. The first tag is an actual countermeasure for the impersonation attack and the second tag is just for adding mutual authentication for *Alice* and *Bob*. Two authentication tags are illustrated in Figure 4.

4.4 Analysis of Countermeasure

As illustrated in Figure 4, despite of adding two authentication tags, the malicious $KDC_{\bar{A}}$ holding password pwa is still able to make messages in the step (5) and to perfectly forge messages M_8', M_9' in the step (8) to go through verification process.

(5) $ID_A, ID_B, m_5', Ticket_B$
(8) $M_8' = g^{r\phi}, M_9' = H(ID_A, ID_B, g^{xr\phi}, g^{xr})$

However, on the side of $KDC_B, KDC_{\bar{A}}$ should face an authentication process $E_{R'}(g^{r'})$ for $R' = H(g^{yr'})$ in the step (7) of Figure 4. $KDC_{\bar{A}}$ with pwa never be able to compute $R' = H(g^{yr'})$ satisfying $R' = (D_{pwa}(m_5'))^{r'}$ because m_5' was selected randomly by $KDC_{\bar{A}}$ hence $KDC_{\bar{A}}$ cannot compute y and r'.

Finally, KDC_B is able to let everyone know a fail of the protocol.

V. Concluding Remarks

It has been not only complicated but also prone to error to design a password-based key agreement protocol preserving both security and efficiency. Even though a protocol is designed under the well-known security model and computational assumptions with formal security proof, there are still possibilities that subtle faults can be found in the protocol, as we have shown in this paper. It also reminds that we should much more pay attention when performing a security analysis on a certain protocol.

In this paper, we have pointed out that the claim of insecurity on EC2C-PAKA in [1] is not valid. In addition, we have shown that the improved C2C-PAKA protocol by Xu and Feng has vulnerability against an impersonation attack by a malicious server. A countermeasure of the attack is also discussed.

Reference

[1] D. Feng and J. Xu, "A new client-to-client password-authenticated key agreement protocol," IWCC'09, LNCS 5557, pp. 63-76, Jun. 2009.

[2] J. W. Byun, D. H. Lee, J. I. Lim, "EC2C-PAKA: an efficient client-to-client password-authenticated key agreement," Information Science, vol 177, pp. 3995-4013, Oct. 2007.

[3] J. Black and P. Rogaway, "Ciphers with arbitrary finite domains," RSA Data Security Conference, Cryptographer's Track (RSA CT '02), LNCS 2271, pp. 114-130, Feb. 2002.

[4] H. Chung, W. Ku, M. Tsaur, "Weakness and improvement of Wang et al.'s remote user password authentication scheme for resource limited environments," Computer Standards and Interfaces, vol. 31 pp. 863-868, Jun. 2009.

[5] Han-Cheng Hsiang, Wei-Kuan Shih, "Weaknesses and improvements of the Yoon-Ryu-Yoo remote user authentication scheme using smart cards," Computer Communications, vol. 32, issue 4, pp. 649-652 Mar. 2009.

[6] P. Kocher, J. Jaffe, B. Jun, Differential power analysis, Proc. Advances in Cryptology, CRYPTO'99, pp. 388-397, Aug. 1999.

[7] N. Y. Lee, Y. C. Chiu, "Improved remote authentication scheme with smart card," Computer Standards and Interfaces, vol. 27 issue. 2, pp. 177-180, Jan. 2005.

[8] J. Munilla, A. Peinado, "Off-line password-guessing attack to Peyravian-Jeffries's remote user authentication protocol," Computer Communications, vol. 30, issue 1, pp. 52-54, Dec. 2006.

[9] Binod Vaidya, Jong Hyuk Park, Sang-Soo Yeo, Joel J.P.C. Rodrigues, "Robust one-time password authentication scheme using smart card for home network environment," Computer Communications, vol. 34, issue 3, pp. 326-336, Mar. 2010.

[10] Shengbao Wang, Zhenfu Cao, Maurizio Adriano Strangio, Lihua Wang, "Cryptanalysis and improvement of an elliptic curve Diffie-Hellman key agreement protocol," IEEE Communications Letters, Vol. 12, no. 2, pp. 149-151, Feb. 2008,

[11] Yan-yan Wang, Jia-yong Liu, Feng-xia Xiao, Jing Dan, "A more efficient and secure dynamic ID-based remote user authentication scheme," Computer Communications, vol. 32, Issue 4, pp. 583-585, Mar. 2009.

[12] X.M. Wang, W.F. Zhang, J.S. Zhang, M.K. Khan, "Cryptanalysis and im-

provement on two efficient remote user authentication scheme using smart cards," Computer Standards and Interfaces vol. 29 no. 5, pp. 507-512, Jul. 2007.
〔13〕 Jing Xu, Wen-Tao Zhu, and Deng-Guo Feng, "An improved smart card based password authentication scheme with provable security," Computer Standards and Interfaces, vol. 31, issue 4, pp. 723-728, Jun. 2009.
〔14〕 Her-Tyan Yeh, Hung-Min Sun, Tzonelih Hwang, "Security analysis of the generalized key agreement and password authentication protocol," IEEE Communications Letters, Vol. 5, no. 11, pp. 462-463, Nov. 2001
〔15〕 Muxiang Zhang, Yuguang Fang, "Security analysis and enhancements of 3GPP authentication and key agreement protocol," IEEE Transactions on Wireless Communications, vol. 4, no. 2, pp. 734-742, Mar. 2005.
〔16〕 변진욱, 정익래, 이동훈, "서로 다른 패스워드워드를 가진 사용자간의 패스워드 인증 키 교환 프로토콜," 정보보호학회논문지, 13(1), pp. 27-38, 2003년 2월.
〔17〕 변진욱, "효율적이고 안전한 스마트카드 기반 사용자 인증 시스템 연구," 전자공학회논문지 48권 TC편 제 2호, pp. 105-115, 2011년 2월.

〈著者紹介〉

변 진 욱 (Jin Wook Byun) 정회원
2001년 2월: 고려대학교 전산학과 이학사
2003년 2월: 고려대학교 정보보호대학원 정보보호 전공, 공학 석사
2006년 8월: 고려대학교 정보보호대학원 정보보호 전공, 공학 박사
2006년 11월~2007년 12월: 영국 런던대학교, ISG 박사후 연구원
2008년 03월~현재: 평택대학교 정보통신학과 조교수
〈관심분야〉 사용자 인증, 프라이버시 보호 기술, 데이터베이스 보안, 암호 프로토콜