INFORMATICA, 2008, Vol. 19, No. 1, 3–16

Public Key Authentication Schemes for Local Area Networks

Tzung-Her CHEN
Department of Computer Science and Information Engineering, National Chiayi University
300 University Rd., Chia-Yi City, Taiwan 600, R.O.C.
e-mail: thchen@mail.ncyu.edu.tw

Gwoboa HORNG, Chuan-Sheng YANG
Institute of Computer Science, National Chung-Hsing University
250 Kuo-Kuang Road, Taichung, Taiwan 402, R.O.C.

Received: October 2006

Abstract. The invention of public-key cryptography makes many new network applications, such as electronic commerce (CE), possible. However, the widely used Internet is open and unprotected. Therefore, verifying the legitimacy of an individual's public key is very important. Most of the key authentication schemes require one or more trustworthy authorities to authenticate the key of a user. Consequently, the system security is mainly dependent on the honesty of these third parties. Unfortunately, a security solution in wide area networks (for example, the Internet) often cannot be applied to local area networks directly without any modification. Sometimes, a complete rebuild is necessary, especially for performance criteria consideration. In this paper, we propose two simple key authentication schemes that require no certification authorities for computer systems in local area networks, in which a host is responsible for user authentication and it uses a designated password authentication mechanism.

Key words: security, public key cryptosystems, authentication, key management, certificate, local area network.

1. Introduction

The invention of public-key cryptography solved the problem of secure key agreement in conventional secret-key (symmetric) cryptosystems (Diffie and Hellman, 1976). Moreover, it also provided a method for generating digital signature. With public-key cryptographic technology, it becomes possible to conduct secure electronic commerce over open networks.

However, another key management problem arises. Since public keys are often stored in a public file for convenient access and maintenance, an intruder can easily impersonate any user in the system if he can forge the contents of the public files (actually, this is a much easier way to attack the system than to aim at the cryptographic algorithm). Therefore, the authentication of a user and his key is a very important issue of secure key management in public-key cryptosystems. For example, in (Li *et al.*, 2000), Li *et al.*

pointed out an insider attack on Harn's multisignature scheme (Harn, 1999) with undistinguished signing participant.

The common method for key authentication is to use certificates. A certificate is generally a digital document used to verify the legitimacy of a user's key. The format of a certificate and how certificates are generated may vary with different key authentication schemes. Since a user cannot certify himself, most key authentication schemes need one or more third-party authorities, such as a certification authority (CA) or a key authentication center (KAC), to issue certificates. These authorities are "self-certified", and even if there are multiple authorities for a certificate, no certification can be guaranteed if all authorities conspire together.

Since public keys are usually stored in a public file and thus are vulnerable to active attacks, the authentication of a public key is very important in public-key cryptosystems for security reasons. Many key authentication schemes were proposed, base on the need of a key authentication center (KAC) to authenticate keys. Girault (1991) surveyed various key authentication schemes and defined three levels of trust for key authentication schemes. They are:

(1) Level 1: the KAC can calculate a user's private key.
(2) Level 2: the KAC cannot calculate a user's private key, but it can generate a false certificate without detection.
(3) Level 3: the KAC cannot calculate a user's private key, and it can be proven that the KAC generates false certificates if it does so.

Girault also proposed the concept of self-certified public keys, and classified key authentication schemes into three categories: ID-based schemes (Shamir, 1984), certificate-based schemes, and schemes using self-certified public keys. Later, Laih *et al.* (1994) proposed an improved model of Girault's scheme with the strategy of imposing a contradiction of unequal conspiracy between two different authorities.

Public key authentication in wide area networks (for example, the Internet) cannot often be applied to local area networks, such as the Intranet, Virtual Private Network (VPN), Personal Area Networks (PAN), etc., directly without any modification. In the local-area-network environment of an organization, generally only tens, hundreds or thousands users, much fewer than that of the Internet, are involved in a local area and only one authentication server is responsible for the task of inside authentication. Therefore, a complete rebuild is not avoidable for performance criteria. On the other hand, it is well-accepted that the trust level of public key authentication involved third parties should be as less as possible. Therefore, there are more and more schemes proposed to address this issue.

In 1996, Horng and Yang first proposed a key authentication scheme, HY scheme for short, which is similar to the conventional certificate-based scheme, yet requires no authorities (Horng and Yang, 1996). In HY scheme, the certificate of a public key of a user is generated by combining his password and private key together. The hash value of his password is handed over to the server and stored in the server's verification table. In (Zhan *et al.*, 1999), Zhan *et al.* pointed out that HY scheme suffers from the password guessing attack and an improved version, ZLYH scheme for short, has been further proposed.

Furthermore, Lee and Wu also proposed another enhanced scheme to solve the password guessing attack (Lee and Wu, 2001).

In (Lee *et al.*, 2003), Lee *et al.* demonstrated that the ZLYH scheme does not achieve non-repudiation of user's public key, i.e., forging the user's public key is possible. Hence, Lee *et al.* proposed another key authentication scheme, LHL scheme for short, based on the ZLYH scheme. Similar attempts, providing the service of public key authentication in personal area networks, appeared in (Gehrmann *et al.*, 2002) and (Rabin, 1979), but trusted CA is still required. In 2004, Peyravian *et al.* proposed the schemes of the public key distribution without PKI (Peyravian *et al.*, 2004). Their schemes removed the requirement of a trusted CA to save storage, bandwidth and to reduce the complexity of PKI. They rely on the existence of passwords shared between users and service providers. The major problems include (1) the service providers must maintain the passwords of legal users, which will suffer from intrusion attacks, and (2) the passwords must be strong (otherwise, the scheme will suffer from password-guessing attacks) and, hence, are not friendly to use or memorize.

In this paper, we propose two simple key authentication schemes that require no certification authorities for computer systems in local area networks, in which a host is responsible for user authentication and it uses a designated password authentication mechanism. Furthermore, the authors shall show that a fault exists in the latest version, the LHL scheme.

The rest of this paper is organized as follows. In Section 2, we give a brief review of some fundamental theorems, password authentication mechanisms, and public-key cryptosystems. In addition, the LHL scheme is briefly described and a fault is shown. In Section 3, we present a key authentication scheme for cryptosystems based on discrete logarithms. In Section 4, we present a key authentication scheme for RSA cryptosystem. Finally, conclusions are given in Section 5.

2. Preliminaries

2.1. *Password Authentication Mechanisms*

In a multi-user computing system, a password authentication mechanism is often used to authenticate legal users. A password is a string of characters assumed to be known only to the system and its user. When a user wants to login to the system, he first enters his user identification, $user\text{-}id$, and then his password, pwd, in response to the system's request. The system then verifies whether the submitted pair $(user\text{-}id,\ pwd)$ is legal or not.

A straightforward implementation of the verification process is that the system keeps all legal pairs of $(user\text{-}id,\ pwd)$ in a table. However, once an intruder gains access to the table, the system becomes insecure. Therefore, to protect against searching for a valid password, a cryptographic solution was proposed (Evans *et al.*, 1974), which uses a one-way function $f(\cdot)$ for mapping a password to its image, and every user's password image is stored in the password table instead of the actual password.

Based on the discrete logarithm problem, we can choose a large prime p and a primitive element $g \bmod p$, and use the function $g^{pwd} \bmod p$ for the image of every user's password pwd, so that these images can be stored in a publicly accessible password table, without revealing contents of the real passwords. Practically, the size of p is recommended to be 512-bit for security in the current time. We note that in the password authentication mechanism of most UNIX systems today, a password is used as the encryption key to the crypt function, which uses 25 rounds of modified DES to encrypt a string of 0s, and the result is 56-bit. Details can be found in (Morris and Thompson, 1979).

2.2. *Cryptosystems Based on Discrete Logarithms*

There are many cryptosystems based on discrete logarithms, such as ElGamal's scheme (ElGamal, 1985) and DSA (NIST FIPS, 1992). In such schemes, there is a large prime p and a primitive element $g \bmod p$ shared among a group of users. Each user randomly generates an integer $x < p$, and then calculates $y \equiv g^x$. The public key of the user is y and the private key is x.

ElGamal's scheme can be used for both encryption and digital signatures. They are reviewed in the following. To encrypt a message M, compute $c_1 \equiv g^k \bmod p$, and $c_2 \equiv My^k \bmod p$ where k is a randomly chosen integer relatively prime to $p-1$. The pair (c_1, c_2) is the ciphertext. To decrypt (c_1, c_2) for the original message M, compute $M \equiv c_2/c_1^x \bmod p$.

To sign a message M, compute $r \equiv g^k \bmod p$, where k is a randomly chosen integer relatively prime to $p-1$. Then solve for s in the following equation: $M \equiv xr + ks \bmod p-1$. The signature for M is the pair (r, s). To verify a signature, confirm that $g^M \equiv y^r r^s \bmod p$.

ElGamal's signature scheme can be extended to sign two messages simultaneously. To sign two messages M_1 and M_2 in the same time, compute $r \equiv g^k \bmod p$, where k is a randomly chosen integer relatively prime to $p-1$. Then solve for s in the following equation: $M_1 \equiv xM_2r + ks \bmod p$. The signature for M_1 and M_2 is the pair (r, s). To verify a signature, confirm that $g^{M_1} \equiv y^{rM_2} r^s \bmod p$.

DSA (Digital Signature Algorithm) is proposed by the NIST. It is a variant of the ElGamal's signature scheme with a smaller signature size (of 320 bits, while ElGamal's scheme usually produces 1024-bit signatures). Details are described in (NIST FIPS, 1993).

2.3. *Cryptosystems Based on Factorization*

RSA cryptosystem was proposed by Rivest, Shamir, and Adleman in (Rivest *et al.*, 1978). It is a public key cryptosystem that can be used for both data encryption and digital signatures. The security of RSA cryptosystem is based on the difficulty of factoring large numbers. A user randomly choose two large primes p and q, and computes the product $N = pq$. Then the user randomly chooses the encryption key e such that

$GCD(e,(p-1)(q-1)) = 1$, and computes the decryption key d such that $de \equiv 1 \bmod ((p-1)(q-1))$. The public key is the pair (e, N), and the private key is d.

To encrypt a message M, compute the ciphertext $c \equiv M^e \bmod N$. To decrypt the ciphertext c, compute $M \equiv c^d \bmod N$.

To sign a message M, compute the signature $s \equiv M^d \bmod N$. To verify the signature s on the message M, confirm that $M \equiv s^e \bmod N$.

Another famous scheme based on factorization is the Rabin cryptosystem (Rabin, 1979). Though it seems more secure that if there are multiple certification authorities to issue certificates, it is still possible that a user is impersonated if all authorities conspire together. Moreover, making all authorities work together is really a complicated task. In the following two sections, we propose two key authentication schemes that require no additional authorities under their presumption. They are for different cryptosystems: the first scheme is for cryptosystems based on discrete logarithms, such as ElGamal's scheme and DSA; the second scheme can be generally used for other cryptosystems, such as the RSA cryptosystem and Rabin's cryptosystem.

2.4. *Review and Cryptanalysis of LHL Scheme*

To remedy the password guessing attacks and public key forging attacks of (Zhan *et al.*, 1999; Lee and Wu, 2001; Horng and Yang, 1996), the LHL scheme was proposed.

In the certificate generation stage, suppose each user chooses his private key K_{priv} and password pwd. Hence, the corresponding public key K_{pub} as $K_{pub(A)} \equiv g^{K_{priv(A)}} \bmod p$, where g is a primitive element over $GF(p)$. A one-way function is assumed to be $f(x) \equiv g^x \bmod p$.

Then he does the following operations:

(1) Select a random number r such that the greatest common divisor of $(pwd + r)$ and K_{priv} is equal to 1.
(2) Find two numbers a and b to satisfy the equation $a(pwd + r) + bK_{priv} = 1$.
(3) Compute $f(pwd + r)$ and $R \equiv g^r \bmod p$.
(4) Send $(f(pwd + r), R, a, b)$ to the server in a secure way.
(5) Generate the certificate $C \equiv \frac{pwd+r}{f(pwd+r)+K_{priv}} \bmod p - 1$.
(6) Make C and K_{pub} public over networks.

On the other hand, upon receiving $(f(pwd + r), R, a, b)$, the server first verifies if $f(pwd + r) \equiv f(pwd)R$ and $f(pwd + r)^a K_{pub}^b \equiv g \bmod p$. If the two equations hold, the server stores them in the public password table which cannot be modified or forged by an intruder, for example protected using access control technique.

In the key verification stage, the sender, who wants to communicate with the receiver, does the following operations:

(1) First obtain $(f(PW + r), R, a, b)$ and (C, K_{pub}) of the receiver from the public password table in the server and the public directory in the network, respectively.
(2) Verify the certificate C of the public key K_{pub} of the receiver by checking if

$$f(C) \equiv f(pwd + r)^{aC} K_{pub}^{bC}$$

$$\begin{aligned}
&\equiv g^{a(pwd+r)C} g^{bCK_{priv}} \\
&\equiv g^{a(pwd+r)C+bCK_{priv}} \\
&\equiv g^{C(a(pwd+r)+bK_{priv})} \\
&\equiv g^{C} \bmod p.
\end{aligned}$$

If it holds, the sender accepts the public key K_{pub} of the receiver to encrypt the communication content; otherwise, he rejects the public key.

Unfortunately, there is a fault in the LHL scheme. In the certificate generation stage, the user computes the certificate $C \equiv \frac{pwd+r}{f(pwd+r)+K_{priv}} \bmod p-1$. In the key verification stage, the sender verify the public key according to C by the equation:

$$f(C) \equiv f(pwd+r)^{aC} K_{pub}^{bC}.$$

If replace the certificate C with an alternative number n, the above verification equation still holds as follows:

$$\begin{aligned}
f(pwd+r)^{an} K_{pub}^{bn} &\equiv g^{a(pwd+r)n} g^{bnK_{priv}} \\
&\equiv g^{a(pwd+r)n+bnK_{priv}} \\
&\equiv g^{n(a(pwd+r)+bK_{priv})} \\
&\equiv g^{n} \\
&\equiv f(n) \bmod p.
\end{aligned}$$

Obviously, no matter what the certificate is, the verification equation still holds.

3. A Key Authentication Scheme for Cryptosystems Based on Discrete Logarithms

In this section, the first scheme based on discrete logarithms is proposed. Suppose a user A, whose password is pwd_A, generated a random number $K_{priv(A)}$ as his private key and computed his public key $K_{pub(A)}$ as $K_{pub(A)} \equiv f(K_{priv(A)}) \equiv g^{K_{priv(A)}} \bmod p$.

3.1. *Assumptions*

This scheme is based on the following assumptions:

(1) The password authentication mechanism of the system uses the following exponentiation function for one-way mapping:

$$f(x) \equiv g^{x} \bmod p,$$

where p is a large prime, g a primitive element $\bmod p$, and $f(\cdot)$ is public to all users. By applying this function, the image of A's password pwd_A is stored in the password table as $f(pwd_A \oplus K_{priv(A)})$, in which $\oplus$ is excusive-or operation and $K_{priv(A)}$ is additionally used to resist password guessing attack.

(2) In addition, a public hash function $h_p(\cdot)$ can be used to hash the result of $f(\cdot)$ of a password to a smaller sized image for saving the storage space of system password table. In this case, the image of the password pwd_A is stored in the password table as

$$h_p\big(f(pwd_A \oplus K_{priv(A)})\big).$$

(3) The system's password table can be directly accessed by every user, and it is well-protected by the system so that it can not be modified illegally. (This is reasonable because a password table is so important that, if a user's password does not map to the image in the password table, the user loses his identity in the system, and thus everything.)

(4) Function $f(\cdot)$ is the same exponentiation function used by the cryptosystem based on discrete logarithms.

Based on assumption (3), one may wonder why not let public keys be well-protected like passwords. One reason is that some users in a group may keep public keys in their local storage for convenient access, and the public keys are verified only at need. In this case public keys are vulnerable to attacks.

3.2. *Authentication Processes*

Certificate generation

Now user A can compute the certificate C_A of his public key by combining his password pwd_A and his private key $K_{priv(A)}$ such that

$$C_A \equiv (pwd_A \oplus K_{priv(A)} + K_{priv(A)}K_{pub(A)}) \bmod p - 1.$$

Key verification

We can see that

$$\begin{aligned} f(C_A) &\equiv g^{(pwd_A \oplus K_{priv(A)} + K_{priv(A)}K_{pub(A)}) \bmod p-1} \bmod p \\ &\equiv (g^{pwd_A \oplus K_{priv(A)}} g^{K_{priv(A)}K_{pub(A)}}) \bmod p \\ &\equiv (g^{pwd_A \oplus K_{priv(A)}} \bmod p)(g^{K_{priv(A)}K_{pub(A)}} \bmod p) \bmod p \\ &\equiv f(pwd_A \oplus K_{priv(A)}) K_{pub(A)}^{K_{pub(A)}} \bmod p. \end{aligned}$$

And if the password image is hashed, we get

$$h_p\big(f(pwd_A \oplus K_{priv(A)})\big) \equiv h_p\big(f(C_A)/K_{pub(A)}^{K_{pub(A)}}\big).$$

If user B wants to access user A's public key, he first asks user A for $K_{pub(A)}$ and C_A or get them from a public file, and obtains A's password image $f(pwd \oplus K_{priv(A)})$ or $h_p(f(pwd \oplus K_{priv(A)}))$ directly from the password table. Then user B can verify the

legitimacy of user A's public key by checking if

$$f(pwd_A \oplus K_{priv(A)}) \equiv f(C_A)/K_{pub(A)}^{K_{pub(A)}}, \text{ or}$$
$$h_p\big(f(pwd_A \oplus K_{priv(A)})\big) \equiv h_p\big(f(C_A)/K_{pub(A)}^{K_{pub(A)}}\big).$$

3.3. *Analysis*

Under the assumption that the image of A's password $f(pwd \oplus K_{priv(A)})$ is well-protected by the system so that it can not be modified illegally, what an attacker can do is forging the public key, guessing the password, or deducing the private key.

Forging a public key

In this scheme, trying to forge someone's public key is not an easy task. Suppose that an intruder wants to substitute a false key K_{false} for user A's public key. In order that K_{false} can be verified as a legal public key, the intruder should also substitute a false certificate C_{false} for the original one so that $f(C_{false}) \equiv f(pwd_A \oplus K_{priv(A)})K_{false}^{K_{false}}$ or $h_p(f(pwd_A \oplus K_{priv(A)})) \equiv h_p(f(C_{false})/K_{false}^{K_{false}})$. To find C_{false}, the intruder has to compute

$$C_{false} \equiv f^{-1}\big(f(pwd_A \oplus K_{priv(A)})\big) + f^{-1}\big(K_{false}^{K_{false}}\big) \bmod p-1 \tag{1}$$

or

$$C_{false} \equiv f^{-1}\big(f(pwd_A \oplus K_{priv(A)})K_{false}^{K_{false}}\big) \bmod p-1 \tag{2}$$

or

$$C_{false} \equiv f^{-1}\Big(h_p^{-1}\big(h_p(f(pwd_A \oplus K_{priv(A)}))\big)K_{false}^{K_{false}}\Big) \bmod p-1, \tag{3}$$

where only $f^{-1}(K_{false}^{K_{flase}})$ is controllable by the intruder. Since the intruder does not know user A's password and he cannot change $f(pwd_A \oplus K_{priv(A)})$ or $h_p(f(pwd_A \oplus K_{priv(A)}))$ in the password table, in both Eq. (1) and (2), the intruder has to deal with the discrete logarithm problem, while in Eq. (3), he faces the problem of reversing the hash function first.

On the other hand, in (Lee *et al.*, 2003), Lee *et al.* demonstrated a method to forge the public key successfully on (Zhan *et al.*, 1999; Lee and Wu, 2001; Horng and Yang, 1996). However, it does not work in this proposed scheme. Assume a dishonest legal user, say D, uses his private key $K_{priv(D)}$ to sign a document. Normally, the signature is verified using the public key $K_{pub(D)}$. However, the signer may deny the signature later by choosing a false certificate C_f, not his certificate C_D, to derive the false public key K_f as follow:

(1) Compute $K_f^{K_f} \equiv \frac{f(C_f)}{f(pwd_D \oplus K_{priv(D)})} \bmod p$; and

(2) Try to obtain K_f from $K_f^{K_f} \bmod p$.

However, to solve the above K_f is more difficult than to do the discrete logarithm problem itself (Agnew *et al.*, 1990).

Guessing password

Based on the assumption that the public password table is protected from illegal modification, what an adversary can do in the server end is to obtain $f(pwd_A \oplus K_{priv(A)})$ to further guessing the password pwd_A. Obviously, it is computationally infeasible to guess pwd_A because he must simultaneously guess pwd_A and the private key $K_{priv(A)}$.

In the user end, an adversary can guess the password from the certificate, $C_A \equiv (pwd_A \oplus K_{priv(A)} + K_{priv(A)}K_{pub(A)}) \bmod p-1$, but he also faces the problem to guess the password and the private key. Therefore, the password guessing attack is very difficult for an adversary in the proposed scheme.

Deducing private key

Once the user A's password is compromised for some reasons, an adversary may try to recover the user's private key from $f(pwd_A \oplus K_{priv(A)})$ stored in the server. Clearly, he must be able to solve the discrete logarithm problem.

On the other hand, an adversary may intend to deduce the private key from the certificate $C_A \equiv (pwd_A \oplus K_{priv(A)} + K_{priv(A)}K_{pub(A)}) \bmod p-1$. He performs the following analyses.

He assumes$C_A \equiv a + b \bmod p-1$, where $a \equiv pwd_A \oplus K_{priv(A)}$, and $b \equiv K_{priv(A)}K_{pub(A)}$, and finds all possible pairs (a, b) that satisfy $C_A \equiv a + b \bmod p-1$. For each pair, he tries two steps to obtain the private key as follows.

(1) XOR the compromised pwd_A with a to obtain $\bar{K}_{priv(A)}$. If $\bar{K}_{priv(A)}K_{pub(A)}$ is equal to b, then he goes to the next step; otherwise, goes to the first step to try the next pair.

(2) Verify if $f(\bar{K}_{priv(A)})$ is equal to $K_{pub(A)}$. If it holds, he obtains the private key. Since all numbers from 1 to $p-1$ are the possible candidates for a, there are $p-1$ trial-and-error tests needed. While the prime p is large enough, it is computational infeasible to deduce the private key from C_A.

In the certificate-based schemes or schemes using self-certified public keys, it is still possible that the system becomes insecure if all authorities conspire together. In this scheme, since there are no additional certification authorities for key authentication, it provides no chance to impersonate a user by conspiring with the certification authorities, and thus it seems more likely to be a level 3 scheme defined by Girault.

Unlike ID-based authentication scheme, in which the public key is fixed to the user's ID, our model provides the flexibility for users to change their passwords and private keys. If a user changes his password and(or) public/private key, the three related public information can be easily updated: the user's password image in the system password table, updated by the system, the public key and its certificate, updated by the user himself. Besides, a user can easily carry out the authentication process by himself. Moreover, the scheme can be implemented using self-certified public keys if the hash function $h_p(\cdot)$ is not used during building password images.

For example, in our scheme, user A's public $K_{pub(A)}$ can be calculated by

$$K_{pub(A)} \equiv f(C_A)/f(pwd_A \oplus K_{priv(A)}),$$

where $f(pwd_A \oplus K_{priv(A)})$ is obtained from the password table, and C_A can be treated as the self-certified public key. In this way, the size of system password table will grow up, yet no original public keys are needed to be stored in the public file, and thus half of the original space can be saved, since only certificates, or the self-certified public keys, are left in the file. Like Girault's discussion on schemes using self-certified public keys, if we do not need the cryptographic protocols to be non-interactive, then the public file can even be removed, since one can ask any user for his self-certified public key, and the key itself is a certificate. In this case, the storage space needed is equal to that of ID-based schemes.

4. A Key Authentication Scheme for RSA Cryptosystem

In this section, another scheme for RSA-based cryptosystems is proposed. Suppose user A, whose password is pwd_A, generated parameters for RSA public key cryptosystem and (N_A, e_A) has become his public key.

4.1. *Assumptions*

This scheme is based on the following assumptions:

(1) The password authentication mechanism of the system uses the following exponentiation function for one-way mapping:

$$f(x) \equiv g^x \bmod p,$$

where p is a large prime, g a primitive element $\bmod p$, and $f(\cdot)$ is public to all users. By applying this function, the image of a password pwd_A is stored in the password table as $f(pwd_A \oplus d_A)$, where d_A is user' private key.

(2) The system's password table can be directly accessed by every user, and it is well-protected by the system so that it can not be modified illegally.

(3) Because the size of p is usually smaller then that of the parameter N for the RSA cryptosystem, a hash (message digest) function $h(\cdot)$ is used to hash parameters for the RSA cryptosystem to their hash values with a size smaller than that of p.

4.2. *Authentication Processes*

Certificate generation

The certificate $(C_{1(A)}, C_{2(A)})$ of user A's RSA public key is computed as A's digital signature on the hash value of the public key, using ElGamal's digital signature scheme for two messages. First, user A computes $C_{1(A)} \equiv g^{k_A} \bmod p$, where k_A is a randomly chosen integer relatively prime to $p-1$. Then A hashes his public key to the hashed key $(h(N_A), h(e_A))$ using the hash function $h(\cdot)$, and solves for $C_{2(A)}$ in the following equation: $h(N_A) \equiv (pwd_A \oplus d_A)h(e_A)C_{1(A)} + k_A C_{2(A)} \bmod p-1$.

Now user A has the pair $(C_{1(A)}, C_{2(A)})$ as the certification of his public key.

Key verification

If user B wants to access user A's public key, he first asks user A for (N_A, e_A) and $(C_{1(A)}, C_{2(A)})$ or get them from a public file, and he also gets A's encrypted password $f(pwd_A \oplus d_A)$ directly from the system password table. Then user B hashes A's public key to get $(h(N_A), h(e_A))$, and verifies the legitimacy of A's public key by confirming that $g^{h(N_A)} \equiv f(pwd_A \oplus d_A)^{h(e_A)C_{1(A)}} C_{1(A)}^{C_{2(A)}} \bmod p$.

4.3. *Analysis*

In this scheme, for example, the certificate $(C_{1(A)}, C_{2(A)})$ of user A's RSA public key (N_A, e_A) is computed as A's digital signature on the hashed public key $(h(N_A), h(e_A))$ using the ElGamal's digital signature scheme for two messages. User A's $pwd_A \oplus d_A$ and its image $f(pwd_A \oplus d_A)$ are used as his private key and public key in ElGamal's system.

Forging public key

If an intruder wants to substitute a false public key (N_{false}, e_{false}) for user A's public key, in order that the false key could be verified as legal, he needs to find a false certificate $(C_{1(false)}, C_{2(false)})$ from either

$$h(N_{false}) \equiv (pwd_A \oplus d_A)h(e_{false})C_{1(false)} + k_{false}C_{2(false)} \bmod p - 1 \qquad (4)$$

or

$$g^{h(N_{false})} \equiv f(pwd_A \oplus d_A)^{h(e_{false})C_{1(false)}} C_{1(false)}^{C_{2(false)}} \bmod p. \qquad (5)$$

In Eq. (4), k_{false} and $C_{2(false)}$ can be arbitrarily chosen, while $C_{1(false)}$ should be equal to $g^{k_{false}} \bmod p$ in order to pass the verification process using Eq. (5). Since the intruder does not know user A's password pwd_A and private key d_A, he cannot determine the value of $C_{2(false)}$, and hence the whole false certificate $(C_{1(false)}, C_{2(false)})$. In Eq. (5), because the intruder cannot change $f(pwd \oplus d_A)$ in the password table, he has to compute $C_{1(false)}$ for a certain $C_{2(false)}$, or vice versa. In both cases, the intruder has to deal with the discrete logarithm problem. It is possible for the intruder to calculate a pair of $h(N_{false})$ and $h(e_{false})$ with the original certificate in Eq. (5), but it is of no help for the intruder to impersonate a user.

Guessing password

Based on the same reason analyzed in Section 3.3, this scheme also resists against password guessing attacks.

Deducing private key

Once the user A's password is compromised, an attack may try to deduce the private key d_A from$f(pwd_A \oplus d_A)$stored in the server. To do so, clearly, he must be able to solve the discrete logarithm problem.

Alternatively, he may try to deduce the private key from the equation $h(N_A) \equiv (pwd_A \oplus d_A)h(e_A)C_{1(A)} + k_A C_{2(A)} \bmod p - 1$. In this equation, he does not know two

parameters: d_A and k_A. First, he guesses k_A and obtains $\bar{d}_A$. Subsequently, he checks whether $e_A\bar{d}_A \equiv 1 \bmod N$. If it holds, he obtains the private key. Since all numbers from 1 to $p-1$ are possible candidates for k_A, it is computational infeasible to deduce the private key, when the prime p is large enough.

The storage space needed in this scheme is much more than that of our first scheme for cryptosystems based on discrete logarithms, because of more space needed for a larger password image in the password table, a larger RSA key size, and twice of the space needed for a certificate. Also, it takes more computation time during the authentication process.

5. Conclusions

Key authentication is a very important problem. There are many ways for finding solutions to it. In this paper we propose two solutions for common cryptosystems without certification authorities. Typically, a certificate represents the certification to both a user's identity and his public key. Our strategy is making a user issue certificate for his public key by giving him the "authority" from something only he knows — his password, and the associated password image in the password table is used to verify his authority. What all we need to guarantee a user's authority is a solid password table for every user. In a multi-user computing system, a user's password represents his identity. For the importance of protecting the password table for the validity of every user's identity, it is reasonable to presume that a password table can not be modified by any user in a normal system. This provides a way for building a secure key authentication scheme without additional authorities. Therefore, our schemes can reduce the key management burden imposed on the local area network of an organization.

We emphasis that our schemes are potentially limited to be used in narrow-ranged LAN computer systems. This is because of the most rigorous assumption in our schemes: a public, directly accessible password table maintained by a host computer in the system is needed (by reason of "to see is to believe"). In an open network system, such as the Internet, direct access is probably not allowed, and there is no guarantee on data integrity. Maybe a possible solution is to modify current communication protocols and OS kernels to comply with system-level access and secure communications between system hosts in the need of looking up remote password tables. However, this would induce many other problems, including how to authenticate remote hosts, how to guarantee the accountability of hosts, system scalability, user privacy, identity association.

In the literature, there are many user authentication schemes and key agreement schemes based on Weil and Tate pairing. With our best knowledge, there is no similar work, i.e., key authentication without requiring certification authorities, based on pairing cryptographic techniques proposed yet. In order to benefit from the more efficiency than that of discrete logarithm or factoring based, the Weil, and Tate pairing-based key authentication schemes will be our future works.

References

Agnew, G.B., R.C. Mullin and S.A. Vanstone (1990). Improved digital signature scheme based on discrete exponentiation. *Electronics Letters*, **26**, 1024–1025.

Deng, R.H., and F. Bao (2003). An improved personal CA for personal area networks. In *IEEE Globcom 2003*. pp. 1–5.

Diffie, W., and M.E. Hellman (1976). New directions in cryptography. *IEEE Transactions on Information Theory*, **22**(6), 644–654.

ElGamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inform. Theory*, **31**, 469–472.

Evans, A., W. Kantrowitz and E. Weiss (1974). A user authentication system not requiring secrecy in the computer. *Communications of the ACM*, **17**(8), 437–441.

Gehrmann, C., K. Nyberg and C. Mitchell (2002). The personal CA – PKI for a personal Area Network. In *Proceedings of IST Mobile & Wireless Communications Summit 2002*. pp. 31–35.

Girault, M. (1991). Self-certified public keys. In *Proceedings of EUROCRYPTO '91*. pp. 490–497.

Harn, L. (1999). Digital multisignature with distinguished signing authorities. *Electronics Letters*, **35**, 294–295.

Horng, G., and C. Yang (1996). Key authentication scheme for cryptosystems based on discrete logarithms. *Computer Communications*, **19**, 848–850.

Laih, S., W.H. Chiou and C.C. Chang (1994). Authentication and protection of public keys. *Computers & Security*, **13**, 581–585.

Lee, W.B., and Y.C. Wu (2001). A simple and efficient key authentication scheme. In *Proceedings of The 18th Workshop on Combinational Mathematics and Computational Theory*. pp. 70–77.

Lee, C.C., M.S. Hwang and L.H. Li (2003). A new key authentication scheme based on discrete logarithms. *Applied Mathematics and Computation*, **139**, 343–349.

Li, Z.C., L.C.K. Hui, K.P. Chow and C.F. Chong (2000). Cryptanalysis of Harn digital multisignature scheme with distinguished signing authorities. *Electronics Letters*, **36**, 314–315.

Morris, R., and K. Thompson (1979). Password security: a case history. *Communications of the ACM*, **22**, 594–597.

NIST FIPS Publication 180 (1993). Secure Hash Standard (SHS). National Institute of Standards and Technology, U.S. Department of Commerce, DRAFT.

NIST FIPS Publication XX (1992). Digital Signature Standard. National Institute of Standards and Technology, U.S. Department of Commerce, DRAFT.

Peyravian, M., A. Roginsky and N. Zunic (2004). Non-PKI methods for public key distribution. *Computers & Security*, **23**, 97–103.

Rabin, M.O. (1979). Digital signatures and public-key functions as intractable as factorization. MIT Laboratory of Computer Science. *Technical Report*, MIT/LCS/TR-212.

Rivest, R., A. Shamir and L. Adleman (1978). A method for obtaining digital signatures and public-key cryptosystem. *Communications of the ACM*, **21**(2), 120–126.

Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO '84*. pp. 47–53.

Zhan, B., Z. Li, Y. Yang and Z. Hu (1999). On the security of HY-key authentication scheme. *Computer Communications*, **22**, 739–741.

T.-H. Chen received the BS degree in Department of Information & Computer Education from National Taiwan Normal University in 1991 and the MS degree in Department of Information Engineering from Feng Chia University in 2001. In 2005, he received his PhD degree in Department of Computer Science from National Chung Hsing University. He has been with Department of Computer Science and Information Engineering at National Chiayi University as assistant professor since August 2005. His research interests include information security, information hiding, multimedia security, digital rights management, security for wireless & mobile networks.

G. Horng received the BS degree in Electrical Engineering from National Taiwan University in 1981 and the MS and PhD degrees from University of Southern California in 1987 and 1992 respectively, all in computer science. Since 1992, he has been on the Faculty of the Institute of Computer Science at National Chung-Hsing University, Taichung, Taiwan, R.O.C. His current research interests include artificial intelligence, cryptography and information security.

C.-S. Yang received the MS degree in Department of Computer Science from National Chung Hsing University in 2001. His research interests include information security, and network security.

Viešojo rakto autentifikacijos schemos lokaliniuose tinkluose

Tzung-Her CHEN, Gwoboa HORNG and Chuan-Sheng YANG

Dėka viešojo rakto kriptografijos išradimo, atsirado galimybė naujai panaudoti telekomunikacijų tinklus, pvz., elektroninei komercijai. Tačiau plačiau naudojamas internetas yra atviras ir neapsaugotas. Todėl norint patikrinti subjekto prieigos legalumą, jo viešasis raktas tampa labai svarbus.

Daugelis rakto autentifikacijos schemų reikalauja vienos ar daugiau patikimų šalių autentifikavimo vartotojo viešojo rakto. Todėl sistemos saugumas pagrindinai priklauso nuo šių trečiųjų šalių sąžiningumo. Deja, saugumo sprendimai dideliuose tinkluose, tokiuose kaip internetas, dažnai negali būti panaudojami lokaliniuose tinkluose be tam tikrų modifikacijų. Kartais reikalingas pilnas neprojektavimas, kai nagrinėjami efektyvumo kriterijai. Šiame straipsnyje mes siūlome dvi paprastas rakto autentifikacijos schemas, kurios nereikalauja sertifikacijos centro lokaliniuose tinkluose, kuriuose vedantis subjektas yra atsakingas už vartotojo autentifikaciją ir naudoja priskirtą slaptažodžio autentifikacijos mechanizmą.