Secure Key Management Scheme for Hierarchical Access Control Based on ECC

Yu-Li Lin\textsuperscript{a,d}, Chien-Lung Hsu\textsuperscript{b,c*}, Tzong-Chen Wu\textsuperscript{a,c}, So-Lin Yen\textsuperscript{d}, Chin-Lung Tseng\textsuperscript{b,c}

\textsuperscript{a} Department of Information Management, National Taiwan University of Science and Technology, Taipei 106, Taiwan, Republic of China (R.O.C.)
\textsuperscript{b} Department of Information Management, Chang Gung University, Tao-Yuan 333, Taiwan, Republic of China (R.O.C.)
\textsuperscript{c} Taiwan Information Security Center at NTUST (TWISC@NTUST), Taipei 106, Taiwan, Republic of China (R.O.C.)
\textsuperscript{d} Ministry of Justice, Investigation Bureau (MJIB), Taipei 231, Taiwan, Republic of China (R.O.C.)

Abstract-- In a key management scheme for hierarchical access control, each higher security class can derive the cryptographic keys of his lower security classes. In 2006, Jang and Wang proposed an efficient key management scheme based on elliptic curve cryptosystems. This paper, however, will demonstrate a compromising attack on Jang-Wang scheme to show that the secret keys of some security classes will be compromised. This paper further proposed an improvement to eliminate the pointed out security leak.

Keywords: Key management; Elliptic curve; Hierarchical Access control, Polynomial interpolation

I. INTRODUCTION

Hierarchical access control is a fundamental problem in computer and network systems. All users in such a system form a user hierarchy and can be assigned into a number of disjoint sets of security classes, say $SC = \{SC_1, SC_2, \ldots , SC_n\}$, which are partially ordered by a binary relation “$\leq$”. In $(SC, \leq)$, $SC_j \leq SC_i$ means that the security level of class $SC_j$ is lower than or equal to the security class $SC_i$. This relation also means that $SC_i$ is the predecessor of $SC_j$ and $SC_j$ is the successor of $SC_i$. In other words, users in $SC_i$ can access the encrypted information held by users in $SC_j$. But the opposite is not allowed. Each security class is assigned a secret key $K_i$ which is used to encrypt/decrypt the confidential file(s). When a user in $SC_i$ would like to retrieve data encrypted by $SC_j$, he should get the right key $K_j$.

In order to resolve the access control problem, Akl and Taylor [1] first proposed a cryptographic key assignment scheme in an arbitrary partial order set (POSET) hierarchy. The drawback in Akl and Taylor scheme is that the size of the public information will grow linearly with the number of security classes. MacKinnon et al. [12] presented an optimal algorithm, called the canonical assignment, to reduce the value of public parameters. However, it is difficult to find an
optimal canonical algorithm. In 1990, Harn and Lin [6] proposed a bottom-up key generating scheme, to instead of using a top-down approach as in the Akl and Taylor and MacKinnon et al.’s schemes. Whenever a new security class is added into or deleted from the user hierarchy, the above schemes can not satisfy the security requirements. That is, all the issued keys should be re-generated.

To overcome dynamic access control problem, a number of schemes have been proposed [2-5,8,9,11,13-17]. In 1992, Chang et al. [2] proposed a key assignment scheme based on Newton’s interpolations method and one-way function. In their scheme, a user with higher security clearance must iteratively perform the key derivation process for deriving the secret key of the user who is not an immediate successor. It is inefficient in the key derivation process. In 2001 and 2002, Wu-Chang [15] and Shen-Chen [13] proposed a cryptographic key assignment scheme to solve the access policy using polynomial interpolations. In their scheme, the central authority (CA) does not need to maintain the security classes’ and the users’ secret keys. That is, any user can freely change his/her secret key for some security reasons. But, there is a security leak inherent in both schemes as described in [7]. That is, an attacker can have access to the information items held by others without following the predefined relation, which violates the security requirement. In 2004, Yang and Li [17] proposed a cryptographic key assignment scheme based on one-way hash function. The cryptographic key of Yang and Li’s scheme is determined by one-way hash functions. Hsu et al. [8], however, pointed out some security flaws of Yang and Li’s scheme to show that the claimed security requirement is violated, since the users can overstep his authority to access unauthorized information. They further proposed two improvements to eliminate the pointed out flaws. However, the Yang-Li and the Hsu et al. schemes cannot perform key updating efficiently. In 2006, Jang and Wang [9] proposed an efficient key management and derivation scheme based on the elliptic curve cryptosystems (it is denoted as the Jang-Wang scheme for short). In Jang-Wang scheme, secret key of each security class can be determined by itself instead of a trusted central authority. An attractive advantage of the Jang-Wang scheme is to solve dynamic key management efficiently and flexibly. It is unnecessary to re-generate keys for all the security classes in the hierarchy when the security class is added into or deleted from the user hierarchy.

However, this paper will demonstrate the security leak inherent in Jang-Wang scheme, which implies their scheme cannot achieve the claimed requirements. Furthermore, we propose an improvement to eliminate the pointed out security leak.

The rest of this paper is sketched as follows. In Section 2, we will review Jang-Wang scheme [9]. In Section 3, we will demonstrate the security leak inherent in Jang-Wang scheme. Finally, we give conclusion in Section 4.

II. OVERVIEW OF JENG-WANG SCHEME

Jeng and Wang proposed an efficient key management and derivation scheme based on the elliptic curve cryptosystem to solve the hierarchical access control problems [9]. Their scheme consists of the initialization, the key generation, and the key derivation phases. In the initialization phase, a central authority (CA) determines all system parameters. In the key generation phase, each security class chooses a pair of points and sends the pair of points to CA. CA extracts the pair of points and constructs the public polynomials of each security class. In the key derivation phase, the predecessor can use its own secret key and the public information related to the successor(s) to derive the decryption key(s) for accessing the authorized file(s). Detailed descriptions of these phases are given below

 Initialization phase – CA randomly chooses a large prime
$p$, and $a, b \in \mathbb{Z}_p^*$ be two parameters satisfying that $4a^3 + 27b^2 \mod p \neq 0$. Let $E_p(a, b)$ be an elliptic curve $y^2 = x^3 + ax + b(\mod p)$ over $\text{GF}(p)$ containing a set of points $(x, y)$’s with $x, y \in \mathbb{Z}_p^*$ and a point $O$ at infinity. Let $G \in E_p(a, b)$ be the base point of order $q$, where $q$ is a large prime. CA also selects an transformation function $\tilde{A} : (x, y) \rightarrow v$ for transforming a point on $E_p(a, b)$ into an integer $v \in \mathbb{Z}_q^*$. Finally, CA publishes $(p, q, \tilde{A}, E, G)$.

**Key generation phase** – Without loss of generality, let $SC = \{SC_1, SC_2, \ldots, SC_n\}$ be a user hierarchy with $n$ disjoint sets of security classes which are partially ordered by a binary relation “$\leq$”. CA determines its secret key $n_{ca} \in \mathbb{Z}_q^*$ and publishes the corresponding public key $P_{ca}$, where $P_{ca} = n_{ca}G$. Initially, each security class $SC_i$ (for $i = 1, 2, \ldots, n$) chooses its own encryption key $K_i$, secret key $n_i \in \mathbb{Z}_q^*$, and the corresponding public key $P_i = n_iG$. The security class $SC_i$ then randomly chooses an integer $k$ and encrypts the point $(K_i, n_i)$ as a ciphertext $(kG_i, (K_i, n_i) + kP_{ca})$, and sends the ciphertext to CA. The central authority CA can use his own secret key $n_{ca}$ to decrypt $(kG_i, (K_i, n_i) + kP_{ca})$ to obtain $(K_i, n_i)$ by the equation $(K_i, n_i) = ((K_i, n_i) + kP_{ca}) - n_{ca}(kG)$. Finally, CA generates a polynomial $f_j(x)$ for each security class $j$ ($j = 1, 2, \ldots, m$) by interpolating the points $(\tilde{A}(n_j P_j), K_j)$’s for all $SC_j < SC_i$.

**Key derivation phase** – When the security class $SC_i$ wants to access the encrypted data held by $SC_j$ where $SC_j < SC_i$, it can use its secret key $n_j$, the public key $P_j$ of $SC_j$ and the public information $f_j(x)$ to derive $K_j = f_j(\tilde{A}(n_j P_j))$.

### III. CRYPTANALYSIS AND IMPROVEMENT OF THE JENG-WANG SCHEME

**A. Compromising Attack on the Jeng-Wang scheme**

In this section, we will show that an adversary who is not a user in a user hierarchy can derive encryption key $K_j$ of a security class by root finding algorithm.

According to Jeng-Wang scheme, each security class $SC_i$ generates its key pair $(n_i, P_i)$, and encryption key $K_i$ and gains the polynomial $f_i(x)$ from CA after the key generation phase. We can precisely see that all predecessors’ secret parameters of a security class $SC_j$ are embedded in its public polynomial $f_j(x)$. When CA adds or deletes some predecessors from $SC_j$, CA updates the public polynomial as $f_j(x)$. For those security classes, which remain as predecessors of $SC_j$, their secrets are still at the same positions of $f_j(x)$. The adversary can try to compute $x$-coordinates of points which are used to construct the public polynomials by solving the equation $f_j(x) - f'_j(x) = 0$. The adversary can first get $\tilde{A}(n_j P_j)$ and then compute $K_j = f_j(\tilde{A}(n_j P_j))$, where $SC_j < SC_i$. In Fig. 1, we suppose the security class $SC_7$ is removed and the new public polynomial for $SC_6$ is $f'_6(x)$. As a result, $(\tilde{A}(n_6 P_1))$ and $(\tilde{A}(n_6 P_3))$ are two roots of the equation: $f_6(x) - f'_6(x) = 0$. Then the adversary can derive $K_6$ by computing $f_6(\tilde{A}(n_6 P_1))$ or $f_6(\tilde{A}(n_6 P_3))$. 
B. Improvement of the Jeng-Wang scheme

The proposed attack is effective on the Jeng-Wang scheme due to the fact that an adversary who is not a user in a user hierarchy can derive encryption key $K_i$ of a security class by root finding algorithm. To eliminate such a security flaw inherent in the Jeng-Wang scheme, we can use a one-way hash function $h(\cdot)$ and a random number $r$ to prevent the secret data $\tilde{A}(n_j P_i)$ from being disclosed. Therefore, the points of a polynomial in the Jeng-Wang scheme can be improved by replacing $(\tilde{A}(n_j P_i), K_j)$ with $(h(r \| \tilde{A}(n_j P_i)), K_j)$. Consequently, if a security class $SC_i$ would like to derive $SC_j$’s encryption key $K_j$, the key derivation equation should be changed to $K_j = f_j(h(r \| \tilde{A}(n_j P_i)))$. From the above-mentioned, it can be seen that the positions of embedded secrets are not fixed and $(r \| \tilde{A}(n_j P_i))$ is protected under the intractability of reversing the one-way hash function.

In the following, we will show that the proposed improvement is secure against the attack demonstrated above. From the demonstrated attack, it can be seen that the adversary attempts to compute $f_i(x) - f'_i(x) = 0$ to obtain zeros at these positions in this difference polynomial. Then, the adversary can use the value as the $x$-coordinate to derive the encryption key. From our modification, it can be seen that the positions in $f_i(x) - f'_i(x) = 0$ vary randomly and the $x$-coordinate of the point, $h(r \| \tilde{A}(n_j P_i))$, is protected by the one-way hash function. That is, the positions of embedded secrets are not fixed any more, and the proposed improvement is secure against our attack. Moreover, the improved scheme is as efficient as the original version, since only one extra hash function and a random number are required.

IV. CONCLUSIONS

In this paper, we demonstrate a security leak inherent in Jeng-Wang scheme [9] to show that their scheme violates the claimed security requirements. The security leak of Jeng-Wang scheme is that the positions of embedded secrets remain unchanged when the public polynomial is updated. That is, any one, even an outsider who does not belong to any security class, can derive the secret keys of some security classes from the public information. Furthermore, we propose an improved scheme to eliminate the security leak inherent in the original version. The proposed improvement is as efficient as the original version.

ACKNOWLEDGMENT

We would like to thank anonymous referees for their valuable suggestions. This work was supported in part by National Science Council under the grant NSC 97-2221-E-182-032 and Taiwan Information Security Center (TWISC), National Science Council under the grants NSC 98-2219-E-011-001.

REFERENCES


