Expert Systems with Applications 37 (2010) 7721–7728

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

An efficient date-constraint hierarchical key management scheme for mobile agents

Tzer-Long Chen [a,*], Yu-Fang Chung [b,*], Frank Y.S. Lin [a]

[a] Department of Information Management, Taiwan University, Taiwan
[b] Department of Electrical Engineering, Tunghai University, Taiwan

ARTICLE INFO

Keywords:
Access control
Key management
Mobile agent
Elliptic curve cryptosystem
Date-constraint key management

ABSTRACT

The core competencies of a mobile agent includes its ability to roam freely in different Internet environments, its ability to adapt itself to various types of online servers depending on the characteristics of said server, and its ability to detect its environment and automatic adaptation to said environment while executing the tasks that are assigned by the users. Because of these competencies, the concept of mobile agent is widely used in many different fields, such as Internet business, wireless communication, and information security technologies. Akl and Taylor (1983) suggested the concept of superkey to resolve the key management issues faced by the mobile agent. Later, Volker and Mehrdad (1998) proposed a tree base mobile agent model for access control. The proposed scheme is based upon the studies of public key and a hierarchical mobile agent model with addition of Elliptic Curve Cryptosystem (ECC). ECC enhances the operational ability of mobile agent model, because of shorter key length and higher efficiency on encryption and decryption. In addition, after a period of time, due to error or change in the user's rights, said user will be forced to log out of the system. At the same time, to prevent a user from continuing to use the old key, the system must keep modifying the key so as to change the original access rights of the key, this action could cause unnecessary error and risks, in addition to the large amount of computations that the system needs to perform. Hence, this paper proposes date-constraint key management scheme, where a date is attached to the key, so as to give a validity period to the key. Thus, key management can be more efficient.

1. Introduction

The popularity of the Internet and widespread use of personal computers has made transmission of information over the Internet a part of our daily life. Ensuring secure transmission of data over open network environment to prevent malicious eavesdropping of the data is an important matter of discussion. In this paper, we integrate the mobile agent, which can autonomously detect its environment and adapt itself to it, with hierarchical structure. The ability of mobile agent to adapt itself to its environment makes it suitable for the current Internet environment. The characteristic of the hierarchical structure have each user holding a different encryption key based on his/her access right. Presently, many businesses and government organizations are using the hierarchical structure along with suitable cryptography related technology to ensure data security. The hierarchical structure used in our work can be integrated with that presently used by businesses and government organizations. In addition, our system sets different rights for different users based on their different transaction needs. The encryption key can ensure that the data will be securely transmitted upon requests. However, when the users under the hierarchical structure decide to log off from the system or switch the authority level for access. The system needs to cancel the right of the previously assigned key so as to avoid the key being illegally used to access the data. But, this could cause the system to spend more on computation to keep the key updated.

To resolve the above issue, this paper proposes to apply date-constraint key management scheme in association with mobile agent structure making the encryption key valid for use only within the date-constraint. In other words, users cannot access data if the keys are no longer valid. Because of this characteristic, the system does not need to keep updating keys. The proposed method aims to enhance the key management of mobile agent structure and make it more efficient.

However, there are many potential risks in accessing data; hence, cryptography technologies are needed for protecting data during transmission. This paper applies the Elliptic Curve Cryptosystem (ECC) to generate the keys. The greatest feature of the ECC is that on the same level of security its key-size is comparatively smaller than that of the currently widely used RSA cryptosystem. For instance, the ECC whose key-size is 160 bit is as secure as the RSA with a key-size of 1024 bit. The smaller key-size

* Corresponding author. Tel.: +886 928639694; fax: +886 423504930.
E-mail addresses: d97725005@ntu.edu.tw (T.-L. Chen), yfchung@thu.edu.tw (Y.-F. Chung).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.04.069

of the ECC allows lower memory requirement and greater execution speed. In addition, mobile agent security related research is also extremely important. In our paper, we will discuss how to protect the system so that the system shall not suffer malicious attacks from unauthorized users. When the mobile agents are executing their tasks on the network, they may communicate and exchange information with each other (Karmouch, 1998; Lin, Ou, & Hwang, 2004).

Below are four examples of security threats which a mobile agent may face (Hohl, 1998):

(1) Unauthorized access to server by a partner.
(2) Attack on server by malicious agents.
(3) Attack on agent by other agents.
(4) Attack on agent by malicious server.

In order to prevent the above stated security threat, this paper applies elliptic curve cryptography to enhance the security of the mobile agent and also to keep it away from malicious attacks.

Following the introduction of our research content and structure, Section 2 introduces the concept of the mobile agent and the advantages it can provide; following, we shall explain the mathematical background of the elliptic curve cryptosystem and time-bound key management. Section 3 explains our proposed scheme. Section 4 contains the security analysis of our proposed scheme. Finally, Section 5 furnishes the conclusions.

2. The related research

2.1. Mobile agent

Mobile agent is the software program that can roam freely in the Internet environment from local host to other remote hosts in a network and executes tasks assigned by its user. It reflects the actual situation during the migration and sends the result back to its user. The migration not only involves transferring of agent codes and result, it also includes transfer of the status of the program. It is therefore suitable for use in distributed and wireless network environments.

In addition, the mobile agent is capable of analyzing a situation, determining a solution, and executing tasks all by itself even if the connection with the target server is not constant. What has been described above indicates that the agent program is designed to be self-managing, self-controlling, and self-resolving, because it can control its behaviors and status and will not be interfered by other systems or users.

Today mobile agents are not only used for distributed computation and data search at remote environments, it is also used in network management and workflow system. Advantages of mobile agent are as follows (Lange & Oshima, 1998):

(1) Reducing network load: The distributed system relies heavily on the transmission medium for exchange of messages, especially when a security protocol is used. However, frequent exchange of messages can cause heavy network traffic. A mobile agent does not require a constant connection with the target server. Its user can encapsulate the instructions before sending it to the target server. The mobile agent initiates interaction and communication with the target server only on its arrival at the server, thus network load is reduced due to lessened communication between mobile agent and server over the Internet.
(2) Decreasing network delay time: Network delay often occurs when large number of systems requiring immediate response is controlled via the Internet. Using mobile agent, these tasks can be transferred to the remote server for execution. As a result, there is no need to maintain a constant connection between the source and the target server. Also, the number of times connections are made is lessened, and thus network delay time is also reduced.
(3) Packaging protocol: When using traditional distributed systems, a fixed protocol is needed for data exchange. However, on different operating systems, each server must set up its own protocol. This is inefficient, and at the same time also proffers security problems. If one of the servers was not quick enough with its update, incompatibility issues or delays can easily occur. When mobile agent is migrating to a remote server, its communication protocol can be packaged for migration, and the same can be re-established later for a network connection. This resolves the connection issue, which could be caused by the data migration.
(4) Adapting to a dynamic environment: Mobile agent is capable of moving freely around different network environment. Therefore, depending on the visited host server's capability, the mobile agent must adjust its own capability. Thus, we say that the mobile agent is able to detect the surrounding environment and automatically adapt to the environment.
(5) Asynchronous and autonomous execution method: In a network environment, a constant connection is often not maintained. Often, due to natural calamity or unknown reason, it may not be possible to connect to the desired server. Thus when the cost of Internet connection is rising and it is not suitable to have a constant connection, the mobile agent can be a useful tool since it completes tasks and returns the result in an asynchronous manner. This can reduce the Internet connection cost and achieve non-simultaneous and spontaneous executions.
(6) Innate heterogeneity: Network computing can be of diverse nature in terms of both hardware and software. The mobile agent is independent from the computer and the network transmission layer. It is like a computer is installed in another computer and they are communicating on the same machine. In this way, it does not matter what the characteristics of the original computer are, it can still connect, interact and exchange information with each other. The mobile agent can relate to any environment. Therefore, a mobile agent is suitable for system integration.
(7) Stability and fault tolerance: Mobile agent can also be used to manage exceptions and this gives it good stability and higher level of fault tolerance.
(8) Expandability: Mobile agent structure allows for great expandability and flexibility in adjustments between the source and the target server.

2.2. Elliptic curve cryptosystem

In 1985, Elliptic Curve Cryptography (ECC) was proposed by Koblitz (1987), Miller (1986). The ECC was able to improve the existing cryptogram systems in terms of having smaller system parameter, smaller public key certificates, lower bandwidth usage, faster implementations, lower power requirement, and smaller hardware processor requirement (Wu, 2005). Therefore, using the Elliptic Curve Cryptography to build a cryptosystem is commendable due to the high security and efficiency (Chung, Lee, Lai, & Chen, 2008) it provides. The mathematic settings of Elliptic Curve Cryptosystem are as described below (Chung et al., 2008; Shieh, 2006).

First, elliptic curves can be divided into two families: prime curves and binary curves. Prime curves (Z_p) are good to use in software application, because it does not require extended bit-fiddling operation, which binary curves require. Binary curves ($GF(2^n)$) are best for hardware application as it require a few logic gates to build a powerful cryptosystems. Second, the variable and coefficients of the elliptic curves are limited to the elements of the finite field. Because of this limitation, it would increase the efficiency of ECC computing operation.

In the finite field Z_p, defined modulo a prime p, an elliptic curve is represented as $E_p(a,b){:}y^2 = x^3 + ax + b(\bmod p)$, where $(a,b) \in Z_p$ and $4a^3 + 27b^2 \bmod p \neq 0$. The condition, $4a^3 + 27b^2 \bmod p \neq 0$, is necessary to ensure that $y^2 = x^3 + ax + b(\bmod p)$ has no repeated factors, which means that a finite abelian group can be defined based on the set $E_p(a,b)$ (Huang, Chung, Liu, Lai, & Chen, 2009). Included in the definition of an elliptic curve, a point at infinity denoted as O is also called the zero point. The point at infinity O is the third point of intersection of any straight line with the curve, so that there are points including (x,y), $(x,-y)$, and O on the straight line.

For points on an elliptic curve, we define a certain addition, denoted "+". The addition rules are given below.

(1) $O + P = P$ and $P + O = P$, where O serves as the additive identity.
(2) $-O = O$.
(3) $P + (-P) = (-P) + P = O$, where $-P$ is the negative point of P.
(4) $(P + Q) + R = P + (Q + R)$.
(5) $P + Q = Q + P$.

For any two points $P = (x_p,y_p)$ and $Q = (x_q,y_q)$ over $E_p(a,b)$, the elliptic curve addition operation, which is denoted as $P + Q = R = (x_r,y_r)$, satisfies the following rules.

$$x_r = (\lambda^2 - x_p - x_q) \bmod p, \quad y_r = (\lambda(x_p - x_r) - y_p) \bmod p, \quad \text{where } \lambda = \begin{cases} \left(\frac{y_q - y_p}{x_q - x_p}\right) \bmod p, & \text{if } P \neq Q, \\ \left(\frac{3x_p^2 + a}{2y_p}\right) \bmod p, & \text{if } P = Q. \end{cases}$$

Given an equation of the form denoted as $E_{23}(1,1){:}y^2 = x^3 + x + 1 \bmod 23$, $a = 1$, $b = 1 \in Z_p$, and $4a^3 + 27b^2 = 31 \bmod 23 \neq 0$, points over the elliptic curve $E_{23}(1,1)$ are shown in Table 1 (Johnson, Menezes, & Vanstone, 2001).

Example 2.1. Let $P = (0,1)$ and $Q = (1,7)$ in $E_{23}(1,1)$. When $P \neq Q$, we must derive λ before calculating $P + Q$, as follows:

$$\lambda = \left(\frac{7-1}{1-0}\right) \bmod 23 \equiv 6$$

So, when $\lambda = 6$, x_r and y_r can be derived as shown below:

$$x_r = (6^2 - 0 - 1) \bmod 23 \equiv 35 \bmod 23 \equiv 12$$
$$y_r = (6(0 - 12) - 1) \bmod 23 \equiv -73 \bmod 23 \equiv 19$$

Thus, $P + Q = R = (12,19)$.

To calculate $2P$, $P = (0,1)$, we must first derive λ as follows:

$$\lambda = \left(\frac{3 \times 0^2 + 1}{2 \times 1}\right) \bmod 23 \equiv \left(\frac{1}{2}\right) \bmod 23 \equiv 12$$

So, when $\lambda = 12$, x_r and y_r can be derived as shown below:

$$x_r = (12^2 - 0 - 0) \bmod 23 \equiv 144 \bmod 23 \equiv 6$$
$$y_r = (12(0 - 6) - 1) \bmod 23 \equiv -73 \bmod 23 \equiv 19$$

Thus, $P + P = 2P = (6,19)$.

P and Q are two points on the elliptic curve cryptosystem, and n is a constant value, such that $P = n \times Q$. If n is very large, then when the two points, P and Q are made public, people who attempt to guess n face some difficulty. This problem is known as the Elliptic Curve Discrete Logarithm Problem (ECDLP) (Guan & Jen, 2005).

2.3. Overview of hierarchy-based key assignment scheme with date-constraint

In 2004, Li proposed a scheme (Li, 2004) where the key is given a validity period by adding date-constraint to the key. The scheme can be divided into five phases:

(1) Initialization phase: CA (Central Authority), which is the trusted third party, will first assign a key to each user, and then calculate the public key of each user in the hierarchy.
(2) Key assignment phase: CA chooses six secret random numbers, and then uses the chosen random numbers to calculate two parameters, a and m, and integrate a date into them. Then, CA chooses another secret parameter k and uses the two previously computed parameters, a and m, to calculate a date-bound warrant W, and parameters r, s and secret key x_j, so that the secret key includes a date and is date-constraint. Finally, CA makes r, s, and W public. The parameters, r, s, and W, and the secret key is calculated in the following manner:

$$r = g^k \bmod p$$
$$s = r^{-1} \times k + x_i \times H(W) \bmod q$$
$$x_j = H(k, ID_j)$$
$$W = (m, \alpha)$$

(3) Key derivation phase: When a user has the right to access another's private key, then the same can use the parameters, r and s, and date-bound warrant W made public by CA to calculate the secret parameter k, the calculations are as follows:

$$k = (s - x_I \times H(W)) \times r + \bmod q$$

Then, the user can use the obtained secret parameter k to calculate the private key. The calculations are as follows:

$$x_j = H(k, ID_j)$$

(4) Key expiration check phase: This phase mainly checks whether the time on the date-bound warrant W is within the validity period. CA first calculates parameter m', as follows:

$$m' = H^{100-a}(a_1) \| H^a(a_2) \| H^{12-b}(a_3) \| H^b(a_4) \| H^{31-c}(a_5) \| H^c(a_6)$$

Judge whether m' is equal to m. If m' is not equal to m, then it means the key is expired, thus user cannot continue to use the key.
(5) Key integrity check phase: This phase mainly verifies the other party's private key, to prove that the same is correct. The verification is as follows:

$$g^x = r^{r^{-1}} \cdot y_i^{H(W)} \bmod p$$

Table 1
Points over the elliptic curve $E_{23}(1,1)$.

(0,1)	(0,22)	(1,7)	(1,16)	(3,10)	(3,13)	(4,0)	(5,4)	(5,19)	(6,4)
(6,19)	(7,11)	(7,12)	(9,7)	(9,16)	(11,3)	(11,20)	(12,4)	(12,19)	(13,7)
(13,16)	(17,3)	(17,20)	(18,3)	(18,20)	(19,5)	(19,18)			

2.4. Overview of Volker and Mehrdad's scheme

Volker and Mehrdad proposed a security scheme (Volker & Mehrdad, 1998), which is based upon a tree structure, to design a secure environment for mobile agents. The proposed scheme is shown as in Fig. 1. The mobile agent structure can be broken into two main branches: static and mutable.

(1) Static branch: The static branch of the agent stores all the static information that would not change during the agent's lifetime. The information includes security policy, group classification, certificates, and access control keys. To prevent mobile agents from malicious attack by hosts, the server will use digital signature on the static branch to ensure the completeness and authenticity of data.
(2) Mutable branch: This branch stores all the dynamic information that is gathered by a mobile agent. For instance, the status or information collected by the agent could be modified by the visited hosts on the network. Therefore, each visited host must digitally sign for the information it modified.

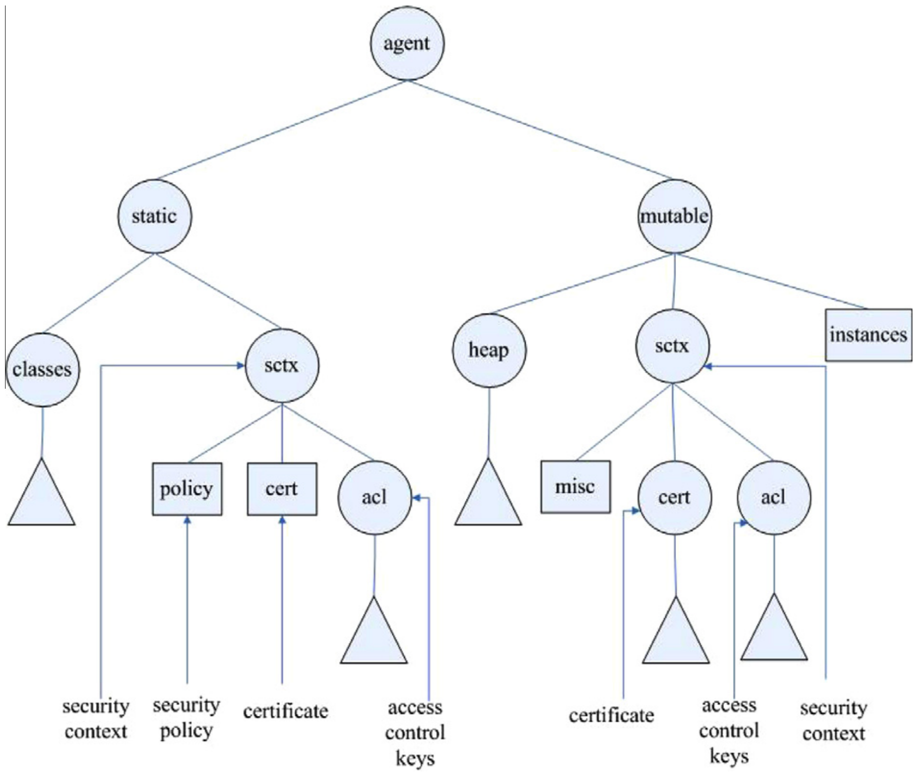

Fig. 1. Framework of mobile agent based on tree structure.

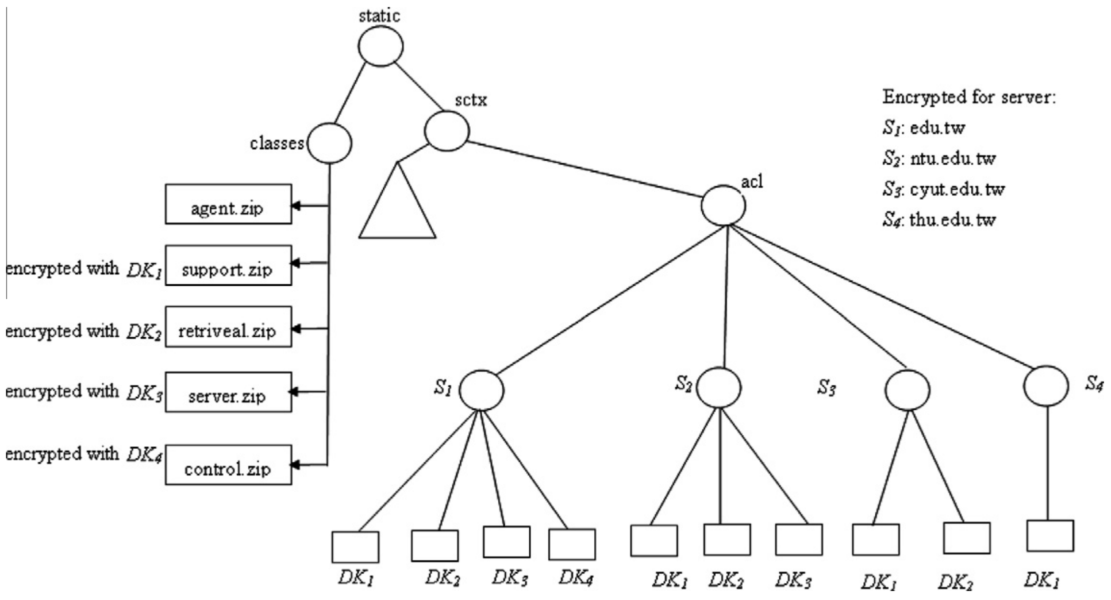

Fig. 2. An example of Volker and Mehrdad's access control and key management.

Because of this signature verification, information comprehensiveness is maintained and the agent is able to build up a trustful relationship with all the hosts it visited.

Fig. 2 is used to illustrate Volker and Mehrdad's agent structure. This figure simply shows the process of how the static branch functions. On the left side of this figure, there are five zipped files: *agent.zip*, *support.zip*, *retrieval.zip*, *server.zip*, and *control.zip*. As the figure shows, *agent.zip* is not encrypted, but the rest of files are encrypted by DK_1, DK_2, DK_3, and DK_4 correspondingly. On the right side, the nodes, S_1, S_2, S_3 and S_4, represent different servers under access control keys. If these servers are authorized to access the specified files, the decryption keys will be copied to the corresponding server folders. For example, S_1 is granted access to all the encrypted files; thus, the keys ($DK_1/DK_2/DK_3/DK_4$) will be copied to the folder of S_1. Based upon the figure's decryption, S_2 is admitted to access the three encrypted files: *support.zip*, *retrieval.zip*, and *server.zip*; therefore, the keys ($DK_1/DK_2/DK_3$) will be duplicated to the folder of S_2. In addition, S_3 is only allowed to access the two encrypted files: *support.zip* and *retrieval.zip*; accordingly, only the keys, DK_1 and DK_2, are in the folder of S_3. Finally, S_4 is permitted to access the first encrypted zipped file, *support.zip*, and then it will just have DK_1 for accessing the designated file.

There are two defects in Volker and Mehrdad's scheme, they are as follows:

(1) A large amount of storage space required inside the mobile agent for storing the generated keys: Upon this scheme, a decryption key can be duplicated to different servers. As shown in Fig. 2, S_1, S_2, and S_3 both have DK_2. Consequently, DK_2 would be continuously duplicated as the number of server increases. This causes wastage of storage space and makes the mobile agent become cumbersome.
(2) Excessive computation on public key generation: Because the decryption keys are repetitively stored under the folder of static/sctx/acl/, the mobile agent user must use more resource and time to compute the public key encryption in order to ensure the security of the folder.

3. Research method

Akl and Taylor recommended an access control scheme (Akl & Taylor, 1983), which is based upon a hierarchical structure model, in 1983. They proposed that each user be assigned to a security group, which can be represented as $C = \{C_1, C_2, C_3, \ldots, C_m\}$. On the basis of a hierarchical structure, this sort of access relationship between one security group and another can be denoted by $C_i \geqslant C_j$. For instance, group of C_i is at a higher level of the hierarchy than C_j when the relationship of $C_i \geqslant C_j$ is declared. This means user of C_i has a greater authority to access the information that is only available to C_j. If the hierarchy network becomes larger, C_i would have to store a lot of decryption keys that are held by the groups at a lower hierarchy. And this would cause a security issue when managing the keys. The proposed solution is to assign one key to each user. Thus, Akl and Taylor came up with the concept of superkey to solve this key management issue. Under the determined hierarchical structure of $C_i \geqslant C_j$, User C_i can use his superkey to obtain C_j's decryption keys through mathematic operations.

Fig. 3 is an illustration of an improved version of Akl and Taylor's structure. Below we shall explain how to access leaf nodes confidential file in a hierarchical structure. filej is the encrypted confidential file; C_i is the internal node, and it also represents a user; skii represents the secret key held by the user. When skii has permission to access some encrypted files, it can obtain confidential file from their corresponding leaf nodes. In Fig. 3, taking node C2 as example, C2 holds secret key sk2, and based on its access right, it can access file1, file2, and file3.

Before the mobile agent is linked to the Internet to execute its assignments, mobile agent user must decide which host would be visited by the mobile agent and what kind of information can be accessed by the visited host. Afterward, he will construct an accessible network in relation to his access policy and gives a different secret key to every internal node. The secret keys are used to encrypt confidential files. A user can use his/her secret key to obtain the confidential file. In order to make access control on a hierarchical structure more effective, the concept of date-constraint

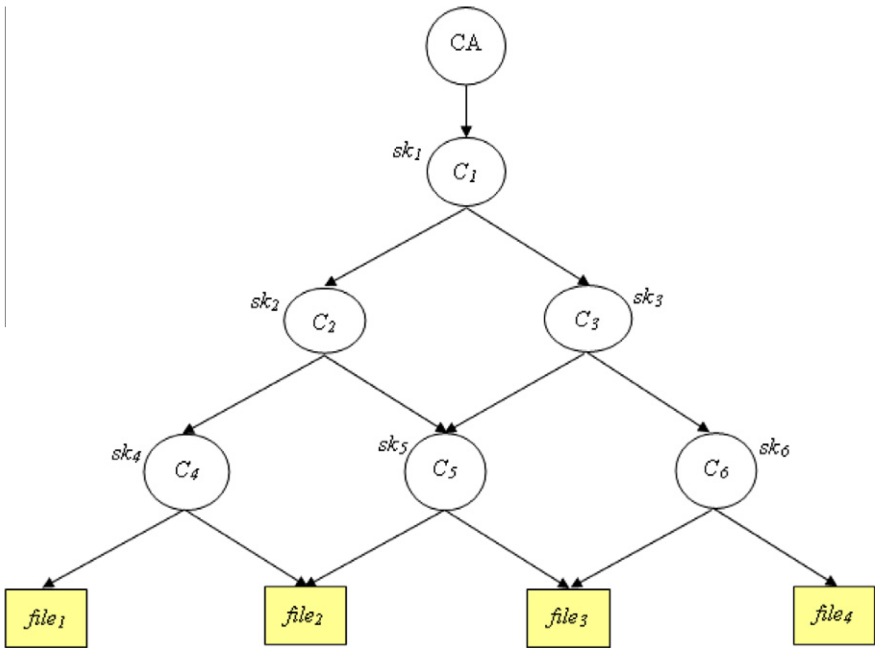

Fig. 3. Structure of decryption keys for mobile agent.

control will be introduced. The main point is to set up a scheme that will allow the user to use his/her existing decryption keys only at a predetermined time lot. In the incorrect time lot, the decryption key holder will be not able to use the key to access the file.

Mobile agent can construct the accessible network through the following steps.

3.1. Initialization phase

CA assigns a private key to each user, and performs the following steps:

Step 1: Define elliptic curve $E_p(a,b)$,where $y^2 = x^3 + ax + b(\bmod p)$. Among which, a and b must satisfy $4a^3 + 27b^2 \neq 0(\bmod p)$, and p is a large prime number.
Step 2: Choose a base point $G = (x,y)$ on $E_p(a,b)$.
Step 3: Suppose there are two users, C_i and C_j, the relation between the two is $C_i \geqslant C_j$, thus the calculations for the public key of C_i and C_j is as follows:

$$pk_i = sk_i G \bmod p$$
$$pk_j = sk_j G \bmod p$$

3.2. Key assignment phase

Then CA performs the following steps for class C_j.

Step 1: Randomly choose six random numbers, v_1, v_2, v_3, v_4, v_5, and v_6.
Step 2: Generate $j = j_1\|j_2\|j_3\|j_4\|j_5\|j_6$, where $j_1 = H^y(v_1)$, $j_2 = H^{100-y}(v_2)$, $j_3 = H^m(v_3)$, $j_4 = H^{12-m}(v_4)$, $j_5 = H^d(v_5)$, and $j_6 = H^{31-d}(v_6)$; y, m, and d, represents year, month, and date, respectively, of the date of expiry, and "$\|$" is the concatenation operator.
Step 3: Calculate $t = H^{100}(v_1)\|H^{100}(v_2)\|H^{12}(v_3)\|H^{12}(v_4)\|H^{31}(v_5)\|H^{31}(v_6)$.
Step 4: CA chooses another secret random number k, and uses the previously calculated parameters t and j to calculate signature date-bound warrant $W = (t,j)$, and also the following public parameters, r, s, R, and the private key x_j of user $C_j x_j$. The calculations are as follows:
$R = k \times G = (x_1,y_1)$ and
$r = x_1 \bmod p$
$s = k^{-1} \times (H(W) - sk_i \times r) \bmod p$, sk_i is the private key of C_i

$$sk_j = H(k, ID_j)$$

where ID_j is the public unique code of identity of C_j.

3.3. Key derivation phase

If $C_i \geqslant C_j$, then C_i can obtain C_j's private key in the following manner:

Step 1: Use public parameter r, s, and W to calculate secret parameter k, as follows:
$k = ((s + sk_i \times H(W)) \times r \bmod p)^{-1}$, sk_i is the private key of C_i.
Step 2: Calculate C_j's private key $sk_j = H(k, ID_j)$

3.4. Key expiration check phase

Step 1: Calculate $t' = H^{100-y}(v_1)\|H^y(v_2)\|\ H^{12-m}(v_3)\|H^m(v_4)\|H^{31-d}(v_5)\|H^d(v_6)$.
Step 2: If t' is not equal to t, then the key is already expired.

3.5. Key signature check phase

Use the following equation

$$R = k \times G = (x_1, y_1) \quad \text{and}$$
$$r = x_1 \bmod p$$
$$s = k^{-1} \times (H(W) - s_i \times r) \bmod p$$

Calculate $V_1 = r \times pk_i + s \times R$ and $V_2 = H(W) \times G$.
Judge whether V_1 is equal to V_2.
If the two are equal, then the signature is true.

Proof

$$\begin{aligned}
V_1 &= r \times pk_i + s \times R \\
&= r \times pk_i + (k^{-1} \times (H(W) - sk_i \times r) \\
&\quad \times R \quad (\because s = k^{-1} \times (H(W) - sk_i \times r) \bmod p) \\
&= r \times pk_i + (k^{-1} \times (H(W) - sk_i \times r \times (k \times G) \quad (\because R = k \times G) \\
&= r \times pk_i + (k^{-1} \times (H(W) - pk_i \times r) \times k \quad (\because pk_i = sk_i G \bmod p) \\
&= r \times sk_i G + k^{-1} \times (H(W) - pk_i \times r \times k \\
&= k^{-1} \times (H(W) \times R) \\
&= k^{-1} \times (H(W) \times k \times G) \quad (\because R = k \times G) \\
&= H(W) \times G \\
&= V_2 \qquad \square
\end{aligned}$$

Example 3.1. Take the hierarchical structure in Fig. 3 for instance, suppose the date of expiry of the key is 98/2/20, and there are two users, C_2 and C_4, whose relation is $C_4 \leqslant C_2$, then CA will follow the following steps to calculate the key:

Initialization phase:

Step 1: Define the elliptic curve equation as $y^2 = x^3 + x + 1(\bmod 5)$.
Step 2: Choose a base point $G = (4,2)$ on the elliptic curve.
Step 3: Calculate public key $pk_2 = sk_2 G \bmod 5$.

$$pk_4 = sk_4 G \bmod 5$$

Key assignment phase:

Step 1: Randomly choose six random numbers, 25, 96, 15, 85, 20 and 60.
Step 2: $y = 98$, $m = 2$, $d = 20$.
Use the above parameters to calculate j, as follows:

$$j_1 = H^{98}(25),\ j_2 = H^{100-98}(96),\ j_3 = H^2(15),\ j_4 = H^{12-2}(85),$$
$$j_5 = H^2 0(20) \text{ and } j_6 = H^{31-20}(60)$$
$$j = H^{98}(25)\|j_2 = H^{100-98}(96)\|j_3 = H^2(15)\|j_4 = H^{12-2}(85)\|$$
$$j_5 = H^{20}(20)\|j_6 = H^{31-20}(60)$$

Step 3: Calculate $t = H^{100}(25)\|H^{100}(96)\|H^{12}(15)\|H^{12}(85)\|H^{31}(20)\|H^{31}(60)$.
Step 4: CA chooses another secret parameter, 150, and signature date-bound warrant $W = (t,j)$, and calculate public parameters r, s, R and private key sk_4; the calculations are as follows:

$$R = 150 \times G = (x_1, y_1)$$
$$r = x_1 \bmod 5$$
$$s = 150^{-1} \times (H(W) - sk_2 \times r) \bmod 5$$
$$sk_4 = H(150, ID_4)$$

Key derivation phase:
C_2 derives the private key of C_4 in the following procedure:

Step 1: Use public parameters r, s, and W to calculate secret parameter, as follows:

$$\begin{aligned}
&((s + x_i \times H(W)) \times r \bmod p)^{-1}\\
&= ((150^{-1} \times (H(W) - sk_2 \times r) + x_i \times H(W)) \times r \bmod (p)^{-1}\\
&= (150^{-1})^{-1}\\
&= 150
\end{aligned}$$

Step 2: Use the obtained secret parameter to calculate C_4 and private key $sk_4 = H(150,ID_4)$.

Key expiration check phase:
Suppose the current date is 98/3/8, use the following steps to verify the validity of the key.

Step 1: Calculate $t' = H^{100-98}(25)\|H^{98}(96)\|H^{12-3}(15)\|H^{3}(85)\|H^{31-8}(20)\|H^{8}(60)$

$$t = H^{100}(25)\|H^{100}(96)\|H^{12}(15)\|H^{12}(85)\|H^{31}(20)\|H^{31}(60)$$

Step 2: t' is not equal to t, meaning the key is expired.

Key signature check phase:

$$\begin{aligned}
V_1 &= r \times pk_2 + s \times R\\
&= r \times pk_2 + (150^{-1} \times (H(W) - sk_2 \times r)\\
&\quad \times R \quad (\because s = 150^{-1} \times (H(W) - sk_2 \times r) \bmod] p)\\
&= r \times pk_2 + (150^{-1} \times (H(W) - sk_2 \times r)\\
&\quad \times (150 \times G) \quad (\because R = 150 \times G)\\
&= r \times pk_2 + (150^{-1} \times (H(W) - pk_2 \times r) \times k \quad (\because pk_2 = sk_2 G \bmod p)\\
&= r \times sk_2 G + (150^{-1} \times (H(W) - pk_2 \times r) \times k\\
&= 150^{-1} \times (H(W) \times 150)\\
&= 150^{-1} \times (H(W) \times 150 \times G) \quad (\because R = 150 \times G)\\
&= H(W) \times G\\
&= V_2
\end{aligned}$$

If V_1 and V_2 are equal, then the signature is true.

4. Security analysis

In this section, a security analysis is performed to examine whether the proposed scheme is secure or not for practical applications. The analysis focuses upon four types of attack that may impact the system security.

(1) Reverse attack
Reverse attack can be defined as follows. When the relationship $C_j \leqslant C_i$ exists between the internal nodes C_j and C_i or they are at the same hierarchical level, the host C_j could try to use its public key pk_j to derive C_i's public key pk_i and attempt to steal data accessible by C_i.
On a disjoint-entity hierarchy, any two internal nodes can be seen as independent units. For example, C_j does not have the right to access the data that is only available to C_i, because it is very difficult to obtain C_i's public key when CA is generating the function $pk_i = sk_i G \bmod p$ of the public key. Also, the security of key generation is based on the ECDLP problem, which is not easy to solve. Hence, C_j's reverse attack would fail. Therefore, the lower level user can never derive the public key of the upper level one if it does not have sufficient information to answer the equation.

(2) Collusion attack
A collusion attack occurs when the lower level users work together to steal the information only accessible by the upper-level user in an accessible network. In the case of Fig. 3, the user C_5 and C_6 decides to work against C_1 by using the obtained information to steal C_1's public key. However, this kind of attack would fail, because ECDLP would make the public key difficult to decode.
Another form of collusion attack is a joint attack by upper level users in an attempt to steal the data of still upper-level users. However, the attack would not succeed, because the upper-level user information is not embedded in the structure of leaf nodes that only low level leaf node users can access.

(3) External collective attack
The intruder attempts to obtain the public key of the internal node to steal or modify the protected data of published system parameters. This is identified as external collective attack. However, although the users in the hierarchy possess more information than external intruders, it is still difficult for them to derive a user's public key. Therefore, it would be even more difficult for an external intruder who has even less data. Hence, it is impossible for the intruder to succeed, because ECC-based structure will prevent it from happening.

(4) Date alteration attack
Suppose a class C_i attempts to extend the validity of the key by using the overdue encryption or continues to use the original encryption of the key that is in the date-bound warrant $W = (t,j)$.
In our proposed method,

$$\begin{aligned}
&H^{100-y}(v_1)\|H^{y}(v_2)\|H^{12-m}(v_3)\|H^{m}(v_4)\|H^{31-d}(v_5)\|H^{d}(v_6)\\
&\quad = H^{100}(v_1)\|H^{100}(v_2)\|H^{12}(v_3)\|H^{12}(v_4)\|H^{31}(v_5)\|H^{31}(v_6) = t
\end{aligned}$$

where $j_1 = H^{y}(v_1)$, $j_2 = H^{100-y}(v_2)$, $j_3 = H^{m}(v_3)$, $j_4 = H^{12-m}(v_4)$, $j_5 = H^{d}(v_5)$, and $j_6 = H^{31-d}(v_6)$, a user tries to extend the key or continues to use the original encryption key and alters system parameter j and public parameter j_1, j_2, j_3, j_4, j_5, and j_6. However, the expiry date of the key was originally defined by CA, and it is protected by the six secret random numbers, v_1, v_2, v_3, v_4, v_5, v_6, and one-way hash function. Suppose a class C_i tries to extend the expiry year from y to y', and C_i can calculate $j_1' = H^{y'}(v_2)$ from $j_1 = H^{y}(v_1)$. However, C_i cannot calculate $H^{100-y'}(v_2)$. Therefore, no entity shall be able to alter the value of t, and the date-bound warrant W cannot be changed at will. Hence, this type of attack can be avoided.

5. Conclusion

In today's Internet environment, mobile agent has a number of superior advantages. Furthermore, these advantages result from the mobile agent's efficient use of network resources, and it will thus be helpful in improving the efficiency of organizations by reducing various costs. Moreover, mobile agent plays an important role in e-commerce, and its contributive value in applications is expected to increase. However, security problems and threats on mobile agent remains a challenging subject. Thus, steps to minimize security problems, increase system operative speed, and reduce needed storage space, need to be taken for mobile agent. Therefore, a more complete mobile agent security system structure is desirable.

In this paper, we propose a date-constraint key management scheme, which gives a date of expiry to a key such that once a key is past its expiry date, its user will not be able to continue to use the key to access information, making key management in the system more efficient; moreover, on account of elliptic curve cryptography, it can reduce access space of keys and lower key generation calculations. In addition, utilizing elliptic curve cryptosystem to generate the keys makes the mobile agent more secure. Because of the Elliptic Curve Discrete Logarithm Problem (ECDLP), ECC is able to generate keys with more complexity and greater protection. Besides, the scheme which we propose also reduces key generation calculations, and thus lowers system load. Finally, in our security analysis, we analyzed four different probable security attacks in depth. The result shows that the proposed scheme is feasible and can be applied in Internet and is not easily susceptible to attacks by malicious users. Thus, our proposed scheme can protect data that is transmitted through the mobile agent platform.

Acknowledgement

This work was supported partially by National Science Council of Republic of China under Grants NSC 98-2221-E-029-025.

References

Akl, S. G., & Taylor, P. D. (1983). Cryptographic solution to a problem of access control in a hierarchy. *ACM Transactions on Computer Systems, 1*(3), 239–248.

Chung, Y. F., Lee, H. H., Lai, F., & Chen, T. S. (2008). Access control in user hierarchy based on elliptic curve cryptosystem. *Information Sciences, 178*(1), 230–243.

Guan, D. J., & Jen, L. H. (2005). Study and implementation of elliptic curve cryptosystem. Master's thesis, National Sun Yat-Sen University of Technology, Kaohsiung.

Hohl, F. (1998). A model of attacks malicious hosts against mobile agents. In *Proceedings of the 4th workshop on mobile object systems: Secure internet mobile computations, Brussels, Belgium* (pp. 105–120).

Huang, K. H., Chung, Y. F., Liu, C. H., Lai, F., & Chen, T. S. (2009). Efficient migration for mobile computing in distributed networks. *Computer Standards and Interfaces, 31*(1), 40–47.

Johnson, D., Menezes, A., & Vanstone, S. (2001). The elliptic curve digital signature algorithm (ECDSA). *Information Security, 1*, 36–63.

Karmouch, A. (1998). Mobile software agents for telecommunications. *Guest Editorial, IEEE Communications Magazine, 36*(7), 24–25.

Koblitz, N. (1987). Elliptic curve cryptosystems. *Mathematics of Computation, 48*(177), 203–209.

Lange, D. B., & Oshima, M. (1998). *Programming and deploying Java mobile agents with aglets*. Massachusetts, USA: Addison-Wesley Press.

Li, J. H. (2004). Hierarchy-based key assignment scheme with date-constraint. Master Thesis, Feng Chia University, Taichung.

Lin, I. C., Ou, H. H., & Hwang, M. S. (2004). Efficient access control and key management schemes for mobile agents. *Computer Standards and Interfaces, 26*(5), 423–433.

Miller, V. S. (1986). Use of elliptic curves in cryptography. In *Advances in cryptology: Proceedings of crypto '85* (Vol. 218, pp. 417–426).

Shieh, C. W. (2006). An efficient design of elliptic curve cryptography processor. Master's thesis, Tatung University, Taipei.

Volker, R., & Mehrdad, J. S. (1998). Access control and key management for mobile agents. *Computer Graphics, 22*(4), 457–461.

Wu, S. T. (2005). Authentication and group secure communications using elliptic curve cryptography. Doctoral Dissertation, National Taiwan University of Science and Technology, Taipei.