Efficient access control and key management schemes for mobile agents

Iuon-Chang Lin\textsuperscript{a,1}, Hsia-Hung Ou\textsuperscript{b}, Min-Shiang Hwang\textsuperscript{c,*}

\textsuperscript{a}Department of Computer Science and Information Engineering, National Chung Cheng University, Chiai, Taiwan, ROC
\textsuperscript{b}Department of Computer Science, National Chung Hsing University, 250 Kuo Kuang Road, 402 Taichung, Taiwan, ROC
\textsuperscript{c}Department of Management Information System, National Chung Hsing University, 250 Kuo Kuang Road, 402 Taichung, Taiwan, ROC

Received 8 July 2003; received in revised form 24 November 2003; accepted 24 November 2003

Abstract

The mobile agent is a promising trend of technology. It is designed for roaming around the Internet and to achieve the goal of autonomy. However, the Internet is an open and, in many cases, a hostile environment. Thus, how to enable the mobile agent to travel through the Internet in safety is an important issue. Recently, Volker and Mehrdad have proposed a structure that supports key management and access control for mobile agent. However, a large amount of storage for storing the secret keys is required in their scheme. In this paper, we will improve their structure and present two novel methods for key management and access control in a hierarchy. According to our security and performance analysis, our proposed scheme is more suitable for mobile agent environments.

\textsuperscript{*} Corresponding author. Fax: +886-4-22857173.
E-mail addresses: iclin@cs.ccu.edu.tw (I.-C. Lin), mshwang@nchu.edu.tw (M.-S. Hwang).
\textsuperscript{1} Fax: +886-5-2720859.

1. Introduction

Due to the unrivalled popularity of the Internet all over the world, almost everything that has always been done in traditional ways, such as business and documentation, now finds a way or another through the Internet. The construction of the World Wide Web offers the Internet its profound influence over our economy, culture, and society. In order to achieve efficiency and effectiveness, the networked environment allows complex forms of distributed computing. However, the bandwidth for current network communication is limited. The delays caused by the traffic jams on the Internet can lead to serious problems, especially when tight interactions between web software systems around the world are required [13].

The mobile agent is a promising technique to overcome this problem. Mobile agents are software programs that act on behalf of users or other software programs. They can travel over the Internet via some communication paths and return the results to the original users. For instance, a search agent is offered for finding information asked for by a query user on a specific station, and the search results are returned to
the original query user. In summary, a mobile agent has
to meet the following properties [19]:

- It can achieve one or more goals automatically.
- It should be able to collate itself and propagate.
- It should be able to collaborate and communicate
  with other softwares and agents.
- It has a scope of competence.
- It has some evolution states to record the
  computation status.

A mobile agent is developed in open, distributed,
and heterogeneous environments. Therefore, it fits the
different frameworks that are built in Java or other
script languages. Summarily, the mobile agent has
such advantages as low network loads, high network
latency resistance, encapsulation of protocols, asyn-
chrony and autonomy, dynamical adaptation, natural
heterogeneity, robustness, and fault tolerance [15].

Many specialists in this domain are newly focused
on mobile agent technologies with a view to modify
human business activities [17]. However, many secu-

The security threats can be classified into the follow-
ing four types [5].

1. protecting hosts from access by unauthorized
   parties;
2. protecting hosts from attacks by malicious agents;
3. protecting agents from attacks by other agents;
4. protecting agents from attacks by malicious
   hosts.

In addition, some private information that is stored
in the agent may be disclosed or tampered. Therefore,
security is the key pointer to the development of the
mobile agent, no matter whether in electronic com-
merce, mobile computing, or other applications. Please
refer to Ref. [4] for more security analyses if necessary.

When a mobile agent works, it has to make contacts
with its visited hosts or other agents. They may either
be trustworthy or malicious. Therefore, the mobile
agent must follow some proper security policies to
keep itself from possible attacks such as agent damage,
denial of service, invaded privacy, confidential data
loss, harassment, social engineering, logic bombs,
event-triggered attack, compound attack, etc. [4].

Many security techniques, such as authenticating cre-
dentials, access control, code verification, audit log-
ning, encrypted/decryption, etc. can be and have been
applied in mobile agent environments.

Recently, many researches [3,14,20] have been
focused on the security of the mobile agent. Corradi
et al. [3] proposed a mobile agent framework called
SOMA. The framework includes the agent, agent
server, management system, and the security policy.
Karnik and Tripathi [14] also offered a framework
called Ajnta. It has the same functions with SOMA. In
contrast to SOMA and Ajnta, Volker and Mehrdad
[20] proposed a tree-based mobile agent structure
which supports such functions as authentication, key
management, and access control. However, Volker
and Mehrdad’s scheme requires a larger agent size.
In this paper, we shall propose two efficient schemes
for managing keys and controlling access in a hierar-
chical structure [1,6–11]. Our proposed schemes can
enhance the performance of Volker and Mehrdad’s
scheme.

The rest of this paper is organized as follows. Before
describing the proposed scheme, we will intro-
duce Volker and Mehrdad’s scheme and point out the
drawbacks of Volker and Mehrdad’s scheme in Section
2. Next, we shall propose our new schemes for key
management and access control in Section 3. In
Section 4, we will analyze the security and perform-
ance of our schemes. Finally, we shall summarize the
benefits of our proposed schemes in the last section.

2. Description of Volker and Mehrdad’s access
control and key management scheme for mobile
agent

The access control and key management scheme
for mobile agent proposed by Volker and Mehrdad
[20] is a useful model for protecting the contents of a
mobile agent. In the following, we will describe
Volker and Mehrdad’s scheme and point out some
drawbacks in Volker and Mehrdad’s scheme.

2.1. Volker and Mehrdad’s scheme

In this scheme, a tree-based agent structure is
designed for supporting authentication, key manage-
ment, and access control. The basic structure is
illustrated in Fig. 1 [20].
The whole structure can be divided into two branches: the static branch and the mutable branch. The static branch contains all the fixed data of the agent such as class codes, security policies, etc. These data do not change during the lifetime of the agent. In order to resist tampering, the agent’s owner might want to use a digital signature to achieve integrity and authentication. The correctness of the contexts in the static branch can be verified by the hosts that the mobile agent is visiting. Oppositely, the variant data is stored in the mutable branch which includes the instances of the classes, a heap as a general storage space, and the confidential context. Furthermore, when the agent has completed its tasks with a specific host, the state of the agent or the collected data may be modified by the host. To keep the integrity of these contexts, the last visited host has to sign the root node. That supports the verifications for the originator and the visited hosts [2].

On the other hand, the confidential contexts in both the static and mutable branches must be kept secret. The technique of access control is required to avoid unauthorized users from accessing the confidential contexts. Therefore, Volker and Mehrdad also proposed an access control and key management strategy to achieve the objectives. In their strategy, the public-key cryptography and symmetric cryptography are used. In addition, a folder is created for each visited host within the static/sctx/acl folder. Each folder contains the corresponding decryption keys that it has authorized to access the confidential files. If the host can access a specific file, it can find the decryption key in its corresponding folder. Furthermore, the contexts of the folder are also encrypted with the host’s public key and would place the seal results under the static/sctx/acl folder. When the agent arrives at a host, the host can find out its corresponding folder from static/sctx/acl in the agent. Each host only has the capability to decrypt and access its corresponding folder by using the corresponding private key.

Fig. 2 illustrates a simple example of the access control and key management strategy [20]. In this figure, we assume that the folder of classes contains four files which are agent.zip, support.zip, retrieval.zip, and control.zip. The file agent.zip is nonconfidential, but the other files are confidential and are encrypted with the keys $K_1$, $K_2$, and $K_3$, respectively. Furthermore, $S_1$, $S_2$, and $S_3$ are the folders which are on behalf of the hosts cyut.
edu.tw, ccu.edu.tw, and ieee-security.org, respectively. If they have the authorities to access the specific files, the corresponding decryption keys are copied into the folders. For example, the host $S_1$ holds $K_1$, $K_2$, and $K_3$, and thus can decrypt the files support.zip, retrieval.zip, and control.zip. The host $S_2$ holds $K_2$ and $K_3$, and thus it can decrypt the files retrieval.zip and control.zip. Similarly, the host $S_3$ only holds $K_1$, and thus it can only decrypt the file support.zip. Furthermore, the folders of static/sctx/acl/S_1, static/sctx/acl/S_2, and static/sctx/acl/S_3 have to be encrypted by their public keys, respectively. By following and using the strategy of access control and key management, the confidential files can be protected.

2. More public key computation: Because the decryption keys are repeated to store in the folder static/sctx/acl/, the agent owner has to use more public key encryption computation to keep the folder secure.

Ideally, the smaller the size of the mobile agent, the better, and the reason is that a smaller mobile agent can be more easily delivered on the Internet. Furthermore, the computational cost must be as little as possible.

3. Proposed access control and key management schemes for mobile agent

In this section, we shall propose two schemes for managing keys and controlling access in a mobile agent. The first scheme is designed in the top-down approach following the concept from Akl and Taylor’s scheme [1]. The second scheme is designed for more reducing the size of public parameters which are used in the first scheme. The securities of the two proposed schemes are both based on the difficulty of factoring a product into two large primes.
In 1983, Akl and Taylor first proposed a cryptographic solution to control access in a hierarchy. In a hierarchic access control policy, all users are allocated into a number of disjoint sets of security classes $C_1, C_2, \ldots, C_m$. According to the partially ordered hierarchy, a user in class $C_i$ can derive the secret keys of the users in any class $C_j$ that is in the lower security levels; however, the opposite is not allowed. The relationship can be expressed as $C_j \prec C_i$. Akl and Taylor brought up a concept, called superkey, to deal with the key management problem. If the class $C_i$ has a security clearance higher than the class $C_j$, the users in $C_i$ can easily derive the secret key of $C_j$ with their own superkey. According to the concept of Akl and Taylor’s scheme, we design two efficient access control and key management schemes for mobile agent that can overcome the drawbacks of Volker and Mehrdad’s scheme.

Before an agent is put to work over the Internet, the home host has to consider what hosts the agent will visit and what data the visited hosts can access. The agent owner has to decide the traveling route and the access policy. Then, the agent owner encrypts each confidential file with an individual secret key. Some symmetric cryptosystems [18], such as AES, DES, or IDEAL, can be used in the encryption procedure. According to the access control policy, the agent owner assigns a superkey to each host and publishes some public parameters for this agent. Each of the host can use its superkey to derive the secret keys that the host is authorized to access. Therefore, our proposed schemes can guarantee that only the authorized hosts can derive the secret keys that are used to encrypt the confidential files.

Before describing our proposed schemes, we first modify Volker and Mehrdad’s tree-based structure and turn it into a hierarchical structure. The hierarchic structure of our proposed schemes is illustrated in Fig. 3. The leaf nodes in the hierarchy represent the secret keys, DKS, that are used to encrypt the confidential files. The internal nodes in the hierarchy represent the superkeys (SKs) that are held by the corresponding hosts. The superkey can be used to derive the secret keys that are located in its descendant nodes. In this figure, the root node, $N_1$, represents the agent owner who holds the superkey, SK1, and has the authority to derive all the secret keys. The node, $N_2$, holds the superkey, SK2, and it has the ability to derive the secret keys DK1, DK2, and DK3. Similarly, the node, $N_6$, holds the superkey, SK6, and it only has the ability to derive the secret keys DK3 and DK4. The details of our proposed scheme are described as follows.

3.1. Scheme 1

Step 1 The agent owner randomly chooses two large primes, $p$ and $q$, and calculates $n = p \times q$. Both $p$ and $q$ need to be kept secret, and the parameter $n$ is public. Next, the agent owner chooses another secure parameter $K_0$.

Step 2 Furthermore, the agent owner chooses a set of distinct primes $\{P_1, P_2, \ldots, P_m\}$ for all the nodes $\{N_1, N_2, \ldots, N_m\}$ in the hierarchy.

![Fig. 3. The hierarchic structure of our proposed schemes.](image-url)
Step 3 Then, the agent owner calculates the public parameters \( \{ t_1, t_2, \ldots, t_m \} \) for each node by the equation

\[
t_i = \prod_{N_j \not\subset N_i} P_j,
\]

where \( N_j \not\subset N_i \) means that the node \( N_j \) is not the descendent of \( N_i \). Furthermore, \( t_i \) satisfies the following two conditions,

\[
t_i \mid t_j, \text{ if and only if } N_j \leq N_i,
\]

and

\[
\gcd (t_i) \mid t_j.
\]

where \( a \mid b \) means \( b \) is the multiple of \( a \); that is, \( b/a \) is an integer.

Step 4 Next, the agent owner delivers the keys \( K_1, K_2, \ldots, K_m \) to all the nodes in the hierarchy. The key, \( K_i \), can be computed by

\[
K_i = K_0^t \mod n.
\]

Step 5 If a host corresponds to the node, \( N_i \), and keeps the relation \( N_j < N_i \), the host can derive the key of \( N_j \) with its superkey, \( SK_i(K_i) \), as follows:

\[
K_j = K_i^t \mod n = K_0^t \mod n = K_0^t \mod n
\]

Therefore, only a superkey is required to store in the corresponding folder for each host.

3.1.1. A simple example for Scheme 1
Similar to the example in the previous section, the structure of our proposed schemes is illustrated in Fig. 4. \( SK_1, SK_2, \) and \( SK_3 \) represent the three different superkeys that are held by the three hosts, \( H_1, H_2, \) and \( H_3 \), which represent cyut.edu.tw, ccu.edu.tw, and ieee-security.org, respectively. \( DK_1, DK_2, \) and \( DK_3 \) represent the three secret keys that are used to encrypt the three confidential files, support.zip, retrieval.zip, and control.zip, respectively. \( H_1 \) holds \( SK_1 \) and has the authority to access the three confidential files support.zip, retrieval.zip, and control.zip; \( H_2 \) holds \( SK_2 \) and has the authority to...
Fig. 5 shows the example of key assignment in Scheme 1. Assume that the agent owner chooses $p = 37$, $q = 47$, and $K_0 = 18$, and publishes $n = p \times q = 1739$. Then, suppose $P_1 = 3$, $P_2 = 5$, $P_3 = 7$, $P_4 = 11$, $P_5 = 13$, and $P_6 = 17$. The agent owner can derive $t_i$ from Eq. (1), and thus, $t_1 = 1$, $t_2 = 231$, $t_3 = 3315$, $t_4 = 23,205$, $t_5 = 19,635$, and $t_6 = 15,015$. Then, the agent owner assigns the keys to the nodes in the hierarchy and applies from Eq. (4). Therefore, $SK_1 = 18^{t_1} \mod 1739 = 18$, $SK_2 = 18^{t_2} \mod 1739 = 1710$, $SK_3 = 18^{t_3} \mod 1739 = 615$, $DK_1 = 18^{t_4} \mod 1739 = 310$, $DK_2 = 18^{t_5} \mod 1739 = 8$, and $DK_3 = 18^{t_6} \mod 1739 = 1207$. Let the parameters $t_i$ and $n$ be public, and the parameters $p$, $q$, and $K_0$ are kept secret.

When the agent arrives at the allocate host, the host can use its superkey, $SK_j$, and the public parameters to calculate the secret key by Eq. (5). For example, $H_2$ can obtain the secret keys, $DK_2$ and $DK_3$, by calculating $DK_2 = SK_2^{t_2} \mod n$ and $DK_3 = SK_2^{t_3} \mod n$, respectively. Furthermore, if the host does not own the authority, it cannot derive the secret key.

3.2. Scheme 2

In Scheme 1, the size of public parameter $t_i$ will grow dramatically with the increasing number of visiting hosts and confidential files. In order to reduce more the size of public parameters, we propose the second scheme that requires less number of public parameters, and the size of public parameter is independent of the number of visiting hosts and confidential files. The following are the details of the second scheme.

Step 1. The agent owner randomly chooses two large primes, $p$ and $q$, and calculates $n = p \times q$. Both $p$ and $q$ need to be kept secret, and the parameter $n$ is public. Next, the agent owner chooses another parameter $K_0$, where $K_0$ is relatively prime to $n$ and $K_0 \equiv [2, n - 1]$.

Step 2. The agent owner chooses a set of distinct numbers $\{e_1, e_2, \ldots, e_u\}$ for all leaf nodes in the hierarchy, where $e_i$ and $(p - 1)(q - 1)$ are relatively prime.

Step 3. The agent owner uses Euclidean algorithm to calculate $\{d_1, d_2, \ldots, d_u\}$, such that

$$d_i \times e_i \equiv 1 \mod \phi(n),$$

where $\phi(n)$ denotes Euler’s totient function, i.e., $\phi(n) = (p - 1)(q - 1)$. In other words, $d_i = e_i^{-1} \mod \phi(n)$.

Step 4. The agent owner delivers the secret keys, $DK_1$, $DK_2$, $\ldots$, $DK_u$, such that

$$DK_i = K_0^{d_i} \mod n,$$

where $1 \leq i \leq u$. The secret key, $DK_i$, is used to encrypt the confidential file $F_i$.

Step 5. The agent owner delivers the superkeys, $SK_1$, $SK_2$, $\ldots$, $SK_r$ for all internal nodes $N_1$, $N_2$, $\ldots$, $N_r$ in the hierarchy, such that

$$SK_i = K_0^{\prod_{j \in N_i} d_j} \mod n,$$

where $1 \leq i \leq r$. Finally, the agent owner assigns the superkey to its corresponding host.

Step 6. If a host corresponds to an internal node, $N_j$, and keeps the relation $N_j < N_i$, the host can derive the secret key, $DK_j$, of the leaf node, $N_j$, with its superkey, $SK_i$, as follows:

$$DK_j = SK_i^{\prod_{j \in N_i} d_j \times 1_{N_j = N_i}} \mod n$$

$$= \left(K_0^{d_i} \right)^{\prod_{j \in N_i} d_j} \times K_0 \mod n$$

$$= K_0^{d_i} \mod n$$

Fig. 5. The key assignment in Fig. 4 using the Scheme 1.
Similar to Scheme 1, only a superkey is required to store in the corresponding folder for each host. Furthermore, only \( u + 1 \) public parameters are required in this scheme.

### 3.2.1. A simple example for Scheme 2

Considering the same example in Fig. 4., \( H_1 \) holds \( SK_1 \) and has the authority to access three confidential files—support.zip, retrieval.zip, and control.zip; \( H_2 \) holds \( SK_2 \) and has the authority to access the files retrieval.zip and control.zip; \( H_3 \) holds \( SK_3 \) and has the authority to access the file support.zip.

Fig. 6 shows the example of key assignment in our proposed scheme. Similar to the example in Scheme 1, the agent owner chooses two large primes, \( p \) and \( q \), and computes \( n = p \times q \). In addition, the agent owner chooses a parameter \( K_0 \). Then, the agent owner keeps \( p \) and \( q \) secret and publishes \( n \). After that, the agent owner can assign the public parameters \( e_1 \), \( e_2 \), and \( e_3 \) to the confidential files support.zip, retrieval.zip, and control.zip, where \( e_1 \), \( e_2 \), and \( e_3 \) are relatively prime to \( (p-1)(q-1) \) and \( e_1 \), \( e_2 \), and \( e_3 \in [2, n-1] \). The agent owner can derive the corresponding \( d_1 \), \( d_2 \), and \( d_3 \) from Eq. (6). Then, the agent owner computes the secret keys, \( DK_1 \), \( DK_2 \), and \( DK_3 \), from Eq. (7), such that \( DK_1 = K_0 d_1 \mod n \), \( DK_2 = K_0 d_2 \mod n \), and \( DK_3 = K_0 d_3 \mod n \). Furthermore, the agent owner can compute the superkeys, \( SK_1 \), \( SK_2 \), and \( SK_3 \), by using Eq. (8), such that \( SK_1 = K_0^{d_1} \times d_2 \times d_3 \mod n \), \( SK_2 = K_0^{d_2} \times d_3 \mod n \), and \( SK_3 = K_0^{d_3} \mod n \).

Suppose that when the agent arrives at \( H_1 \), the host can use its superkey \( SK_1 \) and the public parameters to calculate the secret key by using Eq. (9). Therefore, \( H_1 \) can obtain the secret keys, \( DK_1 \), \( DK_2 \), and \( DK_3 \), such that \( DK_1 = SK_1^{e_2} \times e_3 \mod n \), \( DK_2 = SK_2^{e_1} \times e_3 \mod n \), and \( DK_3 = SK_3^{e_2} \times e_3 \mod n \).

### 4. Security and performance analysis

In this section, we shall analyze the security of our proposed schemes. In addition, we shall also compare our schemes with Volker and Mehrdad’s scheme.

#### 4.1. Security analysis

In this subsection, we will examine the security of the two proposed schemes. The key features of the two proposed schemes are as follows.

1. In Scheme 1, if the secure parameter \( K_0 \) held by the agent owner is disclosed, the system is nonsecure because everyone can derive each superkey and secret key from \( K_i = K_0 t_i \mod n \). However, it is infeasible for anyone to derive \( K_0 \). Any adversary who wants to derive the secure parameter \( K_0 \) by using the known superkey, \( SK_i \), and the modulus \( n \) from \( SK_i = K_0 \mod n \) has to factor \( n \). However, factoring the modulus \( n \) into two large primes is very difficult. So far, there are many factoring algorithms, but they are all time-consuming.
Furthermore, the adversary will try all possible \( K_0 \) until he/she finds the correct one. In Ref. [18], it proves that it is not more efficient than trying to factor \( n \). In Scheme 2, the parameter \( d_i \) must be kept secret. If someone wants to derive the parameter \( d_i \) from \( e_i \) and \( n \), he/she also has to factor the modulus \( n \) into two large primes. This is very difficult as mentioned above.

2. If \( N_j \leq N_i \), it means the host in \( N_j \) does not have the authority to access the secure key in node \( N_i \). In Scheme 1, the public parameters must satisfy the feature: if and only if \( N_j \leq N_i \), then \( t_i \mid t_j \). Therefore, \( t_j/t_i \) is not an integer. The host cannot use its superkey to derive the secret key in \( C_j \) from Eq. (5). Oppositely, if the host in \( N_j \) has the authority to access the secret key in \( N_i \), then \( t_j/t_i \) must be an integer. Therefore, the host can use its superkey, \( SK_i \), to derive the secret key \( DK_j \). In Scheme 2, the superkey, \( SK_i \), has a hidden multiplicative inverse \( d_j \) if \( H_i \) has been authorized to access the confidential file \( F_j \). Therefore, the secret key, \( DK_j \), will be computed by Eq. (9). However, if the host is not authorized, there will be no way for the host to derive the secret key, \( DK_j \), from the superkey, \( SK_i \), or other public parameters.

3. Assume that some hosts want to derive an unauthorized secret key cooperatively by revealing their superkeys. In Scheme 1, because the selection of the public parameters \( t_i \)'s satisfies the feature: \( \gcd_{N_j < N_i} (t_i) \mid t_j \), it is impossible for them to achieve their goal. In Scheme 2, the superkey \( SK_i = K_0^{\prod_{N_j < N_i} d_j} \mod n \) only contains the multiplicative inverses, \( d_j \), of the authorized files. Therefore, the collaboration is not helpful to derive the secret key of the nonaccess file. The collaborative attack cannot work successfully in the two proposed schemes.

4.2. Performance analysis

In this subsection, we shall discuss the performance of our proposed schemes and compare our schemes with Volker and Mehrdad’s scheme. All of these schemes can achieve the goals of access control and key management in a mobile agent.

Because the bandwidth of the current network is limited, how to reduce the communication load is very important. In Volker and Mehrdad’s scheme, some secret keys are stored in the host’s folder repeatedly. For this reason, the size of the agent is increased and occupies more of the network bandwidth. In order to reduce the communication overhead, we must reduce the size of the mobile agent. Instead of storing the secret keys for encrypting the corresponding confidential files, our scheme only stores the host’s superkey in its corresponding folder.

Assume that an agent will visit \( r \) hosts, \( H_1, H_2, \ldots, H_r \), and carry \( u \) confidential files \( F_1, F_2, \ldots, F_u \). Let \( T_1, T_2, \ldots, T_r \) be the number of files that the visited hosts \( H_i \)'s are allowed to access, where \( 1 \leq i \leq r \). Therefore, in Volker and Mehrdad’s scheme, \( \sum_{i=1}^{r} T_i \) secret keys need to be stored in \textit{static/sctx/acl}. However, in both of our proposed schemes, only \( r \) superkeys need to be stored in \textit{static/sctx/acl}.

If the length of the keys is 512 bits in the used symmetric cryptosystem and public key cryptosystem, then, Volker and Mehrdad’s scheme requires \( 512 \times (\sum_{i=1}^{r} T_i) \) bits of memory space to store the keys, but only 512 \( r \) bits of memory space are required in our schemes. Note that the range of \( T_i \) is from 1 to \( u \), i.e., \( 1 \leq T_i \leq u \). Furthermore, in our proposed schemes, the system must maintain some public parameters. In Scheme 1, the system must maintain \( r + u + 1 \) public parameters. In Scheme 2, system only maintains \( u + 1 \) public parameters. Obviously, the number of public parameters is significantly reduced.

On the other hand, each host’s folder in \textit{static/sctx/acl} must be kept secret, and thus, it needs to be encrypted with the host’s public key by using the public key cryptosystem such as RSA. Suppose that the encrypt and decrypt procedure requires one exponential operation in a cryptosystem. Because the plaintext length must be the same or smaller as the length of the public key, Volker and Mehrdad’s scheme requires \( \sum_{i=1}^{r} T_i \) exponential computations to encrypt the secret keys and requires \( \sum_{i=1}^{r} T_i \) exponential computations to decrypt the secret keys. Totally, \( 2 \times \sum_{i=1}^{r} T_i \) exponential computations are required in Volker and Mehrdad’s scheme. However, in both of our proposed schemes, only \( 2 \times r \) exponential computations are required to encrypt and decrypt the superkeys. Furthermore, our schemes require one exponential computation to derive the secret key from its superkey and, thus, it requires \( \sum_{i=1}^{r} T_i \) exponential computations to derive all the used secret keys.
The technique of mobile agent is the star of the near future in electronic commerce, but the security threats are the major obstacles to overcome. In this paper, we have proposed two new key management and access control schemes to protect the content of the agent. Unlike Volker and Mehrdad’s tree-based agent structure, we have designed a new agent structure in a hierarchy. In the two proposed schemes, the agent structure can be easily established according to the agent owner’s access policy. In our proposed schemes, Scheme 2 is more efficient that Scheme 1. It substantially reduces the number of required public parameters that is used in the mobile agent. In particular, both of our proposed schemes can reduce the agent size and required exponential computations. Therefore, our proposed schemes are more suitable for mobile agent environments.

5. Conclusions

In general, the visited hosts are allowed to access more confidential files. Therefore, our scheme is more efficient than Volker and Mehrdad’s scheme.

Acknowledgements

The authors wish to thank many anonymous referees for their suggestions to improve this paper.

References

I.-C. Lin received his BS degree in Computer and Information Sciences from Tung Hai University, Taichung, Taiwan, Republic of China in 1998, and received his MS degree in Information Management from Chaoyang University of Technology, Taiwan in 2000. He is currently pursuing his PhD degree in Computer Science and Information Engineering from National Chung Cheng University, Taiwan. His current research interests include electronic commerce, information security, cryptography, and mobile communications.

H.-H. Ou received his BS and MS degree in Information Management from Chaoyang University of Technology, Taiwan, Republic of China in 1999 and 2001. He is currently pursuing his PhD degree in Computer Science from National Chung Hsing University, Taiwan. His current research interests include mobile agent, information security, and cryptography.

M.-S. Hwang was born on August 27, 1960 in Tainan, Taiwan, Republic of China (ROC.). He received his BS in Electronic Engineering from National Tsing Hua Institute of Technology, Taipei, Taiwan, ROC in 1980, his MS degree in Industrial Engineering from National Tsing Hua University, Taiwan in 1988, and his PhD degree in Computer and Information Science from National Chiao Tung University, Taiwan in 1995. He also studied Applied Mathematics at National Cheng Kung University, Taiwan from 1984 to 1986. Dr. Hwang passed the National Higher Examination in the field of “Electronic Engineer” in 1988. He also passed the National Telecommunication Special Examination in the field of “Information Engineering” and was qualified as advanced technician, first class, in 1990. From 1988 to 1991, he was the leader of the Computer Center at Telecommunication Laboratories (TL), Ministry of Transportation and Communications, ROC. He was also the chairman of the Department of Information Management, Chaoyang University of Technology (CYUT), Taiwan, from 1999 to 2002. He was a professor and chairman of the Graduate Institute of Networking and Communications, CYUT, from 2002 to 2003. He obtained the 1997, 1998, 1999, 2000, and 2001 Distinguished Research Awards of the National Science Council of the Republic of China. He is currently a professor of the Department of Management Information System, National Chung Hsing University, Taiwan, ROC. He is a member of IEEE, ACM, and Chinese Information Security Association. His current research interests include electronic commerce, database and data security, cryptography, image compression, and mobile computing. Dr. Hwang had published 90 articles on the above research fields in international journals.