On the Security of an Efficient Time-Bound Hierarchical Key Management Scheme

Hung-Min Sun, King-Hang Wang, and Chien-Ming Chen
Department of Computer Science,
National Tsing Hua University, Hsinchu, Taiwan
E-mail:hmsun@cs.nthu.edu.tw, {khwang0, kkyy}@is.cs.nthu.edu.tw

Abstract

Recently, Bertino *et al.* proposed a new time-bound key management scheme for broadcasting. The security of their scheme is planted on the hardness breaking of elliptic curve discrete log problem, HMAC, and tamper-resistance devices. They claimed that as long as the three assumptions hold, their scheme is secure. By means of secure, users cannot access resources that they are not granted, even if users collude. In this paper, we demonstrate that this scheme is insecure against the collusion attack. We also provide some possible amendments to this scheme.

Index Terms

Time-bound Key Management, Cryptanalysis.

I. Introduction

Time-bound hierarchical key management schemes have been widely discussed recently [1], [2], [3]. These schemes control different sets of users from accessing different resources at a fixed time of period. Usually, resources are grouped into some classes, and classes have hierarchical relationships amount themselves. Users subscribe to a class can also access the resources in its descendant classes, but not vice versa. Also, users are not allowed to access the resources out of their subscription periods. These schemes efficiently control users access rights with little transmission overheads.

Very recently, Bertino *et al.* [3] proposed another new time-bound scheme using elliptic-curve cryptography. Bertino *et al.* claimed that their scheme is efficient and secure against several attacks. Unfortunately, we found that their scheme is not secure as they claimed. In this paper, we demonstrate that Bertino *et al.*'s scheme is vulnerable to the collusion attack and propose some possible improvements for their scheme.

II. Reviewing Bertino *et al.*'s Scheme

We briefly review Bertino's scheme [3] in this section. Their scheme includes the following phases: the initialization, the encrypting key generation, the user subscription, the decryption key derivation.

A. Initialization Phase

The vendor first defines the partially ordered classes C_i of the data source for $1 \leq i \leq w$. We use the notations $\prec$ or $\preceq$ to represent partial order relations. If $C_j \preceq C_i$ or $C_j \prec C_i$, the user who are in the class C_i can also access all resources in C_j. $C_j \prec C_i$ is different from $C_j \preceq C_i$ only when $C_i \prec C_j$ does not hold. The detail definition of partial order relation is described in [3].

Besides, the vender selects an elliptic curve E over a finite field $\mathbb{F}_q$, a point $Q \in E(\mathbb{F}_q)$ with a large prime order p, integers n_i and g_i for $1 \leq i \leq w$ where $n_i g_j$ are all different modulo p, two random integers a and b, and a keyed HMAC $H_K(-)$ where $H()$ denotes a hash function and K denotes the system's master key which is only known to the vender.

The vendor then computes the following for each classes i:

1) $P_i = n_i Q$ on $E(\mathbb{F}_q)$,
2) h_i where $g_i h_i \equiv 1 (mod\ p)$,
3) the class key $K_i = g_i P_i$ for class C_i,
4) the points $R_{i,j} = g_i K_j + (-K_i)$ where $C_j \prec C_i$.

The values ID_i, $R_{i,j}$ are published and P_i, n_i, g_i, h_i, a and b are kept secret. Once $R_{i,j}$ are released, the class partial order relationships cannot be changed without resetting all keys.

B. Encrypting Key Generation Phase

In this phase, the vendor generates the secret key $K_{i,t}$ for C_i at the time granule $t \in [1, Z]$. $K_{i,t}$ is calculated as (1).

$$K_{i,t} = H_K((K_i)_Y \oplus H^t(a) \oplus H^{Z-t}(b) \oplus ID_i). \quad (1)$$

Note that $(K_i)_Y$ means the $y-$coordinate of the point K_i, $\oplus$ denotes the bitwise XOR operation and ID_i is the public identity of C_i. The notation $H^x(a)$ is defined as $H(H^{x-1}(a))$ for $x \geq 2$ with $H^1(a) = H(a)$.

C. User Subscription Phase

If a user u subscribes to C_i with the time interval $[t_1, t_2]$, the vendor calculates $H^{t_1}(a), H^{Z-t_2}(b)$. The vendor then issues u a tamper-resistant device, like a smart card, which stores H, K, E, $\mathbb{F}_q$, ID_i, h_i, $H^{t_1}(a)$ and $H^{Z-t_2}(b)$. This tamper-resistant device is also embedded with a secure clock. We assume that no unauthorized read or write can be made on the variables it stores. Besides, the class key K_i is distributed to u through a secure channel.

D. Decrypting Key Derivation Phase

Suppose a user u subscribes to the class C_i in the time interval $[t_1, t_2]$. User u can access C_j at time t, if $C_j \preceq C_i$ and $t \in [t_1, t_2]$, by computing the access key $K_{i,t}$. He first retrieves $R_{i,j}$ from the public network and then inputs $R_{i,j}$, ID_j, K_i into the tamper-resistant device. The device then computes (2), (3), and (4).

$$K_j = h_i.(R_{i,j} + K_i) \quad (2)$$

$$H^t(a) = H^{t-t_1}(H^{t_1}(a)), \quad (3)$$

$$H^{Z-t}(b) = H^{t_2-t}(H^{Z-t_2}(b)) \quad (4)$$

Finally, by utilizing the above results, it computes the access key $K_{j,t}$ using (5)

$$K_{j,t} = H_K((K_j)_Y \oplus H^t(a) \oplus H^{Z-t}(b) \oplus ID_j). \quad (5)$$

In case that $C_j = C_i$, user u inputs only K_i into the device. The device then computes $K_{i,t}$ using (1) directly.

III. Cryptanalysis on Bertino's scheme

Assume that two users u_1 and u_2 collude together. User u_1 subscribed to C_i with time intervals $[t_{i_1}, t_{i_2}]$, whereas user u_2 subscribed to C_j with another non-overlapping time intervals $[t_{j_1}, t_{j_2}]$. Assume that $C_j \prec C_i$, so they should not be able to access the resources of C_i (but not in C_j) in the time interval $[t_{j_1}, t_{j_2}]$. It means, they should not be able to derive $K_{i,t}$ where $t \in [t_{j_1}, t_{j_2}]$.

By using u_2's device, they calculate a point S such that $\{S\}_Y = \{K_i\}_Y \oplus ID_j \oplus ID_i$. Then they store S into the device as the fraud K_j. Since the device treats it as K_j, it directly calculates $K_{j,t}$ with S as follows:

$$\begin{aligned} K'_{j,t} &= H_K(\{S\}_Y \oplus H^t(a) \oplus H^{Z-t}(b) \oplus ID_j) \\ &= H_K((\{K_i\}_Y \oplus ID_j \oplus ID_i) \oplus H^t(a) \oplus H^{Z-t}(b) \\ &\quad \oplus ID_j) \\ &= K_{i,t} \end{aligned}$$

As a result, they obtain the crafted access key $K'_{j,t}$, which is indeed equal to $K_{i,t}$, from the output of the device. Then they can access C_i at time t using $K_{i,t}$, in which they are not authorized. We shall notice that this attack also works even if $[t_{j_1}, t_{j_2}]$ and $[t_{i_1}, t_{i_2}]$ are overlapped. That means, these two devices can access the encrypted resources at the same time.

The remaining part is to calculate the point S. Suppose that we follow the suggestion of [3] to implement the elliptic curve over a prime field F_p, a finite field of p elements and p is a prime, with the formula $y^2 = x^3 + ax + b \ mod \ p$. Therefore, finding S is equivalent to finding the root of $x^3 + ax + b \ mod \ p$. Although such as S does not always exist, most of case can successfully find S in polynomial time. Other choices of implementation will also lead to similar result. Also, if the device does not validate the point S is on the curve or not, this attack will succeed without doubt.

IV. Possible Improvements

The causes of the vulnerability are due to: 1) the symmetric property of exclusive-or $\oplus$ and 2) the input materials K_i and $R_{i,j}$ are not authenticated. A simple improvement is to replace

the symmetric operator exclusive-or $\oplus$ in (1) by string concatenation $||$. The resulting equation is shown as (6).

$$K_{i,t} = H_K((K_i)_Y||H^t(a)||H^{Z-t}(b)||ID_i). \quad (6)$$

This amendment is lightweight, however we cannot guarantee that it eradicates all other attacks.

The other method is to prove the authenticity of K_i and $R_{i,j}$. The vendor signs K_i and $R_{i,j}$ using digital signature as follows

$$s_i = Sig(K_i||i),\ \ s_{i,j} = Sig(R_{i,j}||i||j)$$

Then, whenever K_i and $R_{i,j}$ are inputted, users respectively provide s_i and $s_{i,j}$ as well. The device aborts the computation if K_i or $R_{i,j}$ are not inputted with valid signatures. Although it incurs extra computation and transmission loads to the system, this amendment makes the scheme provable secure (see appendix). It is because the device only accepts inputs that prepared by the vendor. In addition, K_i are tightly coupled with i and $R_{i,j}$ are bundled with i, j by the signatures, which disallow attackers to forge or replace any of those.

V. Conclusion

There are many time-bound hierarchical key assignment schemes in literature, which apparently seems to work but they have been shown to be insecure. Nevertheless, the very recent Bertino *et al.*'s scheme is shown broken in this paper. We suggest some simple improvements to fix the scheme against the collusion attack.

Acknowledgement

This work was supported in part by the National Science Council, Taiwan, under Contract no. NSC 96-2628-E-007-025-MY3 and NSC96-3114-P-001- 002-Y.

References

[1] W.-G. Tzeng, "A time-bound cryptographic key assignment scheme for access control in a hierarchy," *IEEE Trans. on Knowledge and Data Eng.*, vol. 14, no. 1, pp. 182–188, 2002.

[2] H.-Y. Chien, "Efficient time-bound hierarchical key assignment scheme," *IEEE Trans. on Knowledge and Data Eng.*, vol. 16, no. 10, pp. 1301–1304, 2004.

[3] E. Bertino, N. Shang, and S. Wagstaff, "An efficient time-bound hierarchical key management scheme for secure broadcasting," *IEEE Trans. on Dependable and Secure Comput.*, vol. 5, no. 2, pp. 65–70, April-June 2008.

Hung-Min Sun received his B.S. degree in applied mathematics from National Chung-Hsing University in 1988, his M.S. degree in applied mathematics from National Cheng-Kung University in 1990, and his Ph. D. degree in computer science and information engineering from National Chiao-Tung University in 1995, respectively. He was an associate professor with the Department of Information Management, Chaoyang University of Technology from 1995 to 1999, and the Department of Computer Science and Information Engineering, National Cheng-Kung University from 1999 to 2002. Currently he is a professor with the Department of Computer Science, National Tsing Hua University. Prof. Sun has published over 100 international journal and conference papers. He was the program co-chair of 2001 National Information Security Conference, and the program committee member of 1997 and 2005 Information Security Conference in Taiwan, 2000 Workshop on Internet and Distributed Systems, 2001-2002, 2005 Workshop on the 21st Century Digital Life and Internet Technologies, 1998-1999 2002-2008 National Conference on Information Security, ACISP04, NCS2001, ICS2002, ITRE2005, NCS2007, ICS 2008, ISC 2008 Special Session on AES Subcommittee, SH 2008. His research interests include computation, cryptography, multimedia security, and network security.

King-Hang Wang was born in Hong Kong, China in 1981. He received his B.S. degree in Information Engineering from Chinese University of Hong Kong in 2002. He is now pursuing his doctoral degree in Computer Science from National Tsing Hua University. He is also an IEEE student member since 2006. His research interests include provable security, digital rights management, steganography.

Chien-Ming Chen received his B.S. degree in computer science and information engineering from the Fu-Jen Catholic University in 2000, his M.S. degree in computer science and information engineering from the Fu-Jen Catholic University in 2002. He is currently pursuing his Ph. D. degree in computer science from National Tsing Hua University. His current research interests include provable security, applied cyptography, multimedia security, and digital right management.

APPENDIX

Recall our second proposed amendment, that requires the vendor to sign on the value of K_i and i as (7) and $R_{i,j}$, i, and j as (8).

$$s_i = Sig(K_i||i) \tag{7}$$

$$s_{i,j} = Sig(R_{i,j}||i||j) \tag{8}$$

Whenever the user inputs the value of K_i and $R_{i,j}$, the smart card will verify their signatures s_i and $s_{i,j}$. If any of the two signatures is invalid, the smart card will return `invalid` to the user.

We wish to sketch the security proof of the scheme by constructing a simulator that contains the adversary $\mathcal{A}$ who claims that he can break the scheme. Before that, we will give the adversary model by declaring her capabilities, her goal, and the security assumptions.

ADVERSARY MODEL

Adversary's Capability

We assume that the adversary $\mathcal{A}$ has the freedom to subscribe finitely many subscriptions. For each subscription $\mathcal{A}$ has made, she will receive the corresponding smart card, the secret K_i, and the signature s_i, if the subscription is for class i. We give $\mathcal{A}$ this ability to model collusion attack.

$\mathcal{A}$ can input any message to her smart cards for many times, as long as the total number of inputs is bounded by a polynomial of the security parameter τ. Depends on the inputs, the smart card will reply the corresponding output to $\mathcal{A}$. If the inputs are invalid, the smart card will output `invalid` to $\mathcal{A}$. $\mathcal{A}$ can also control the clock of the smart card. So, the smart card will attempt to generate the key at time t' even if the current time is t. Of course, we will show in the simulator, if t' is not in the subscription period, the smart card would not be able to generate the key at t'. However, $\mathcal{A}$ is disallowed to retrieve the temporary value calculated inside the smart card, like $H^t(a)$, K_j. In other words, the black box is treated as a black box here.

$\mathcal{A}$ is able to compute hash function H and the key HMAC H_K by querying the oracles $\mathcal{H}_1$ and $\mathcal{H}_2$ respectively. If the adversary is able to break the scheme, she should also be able to break the scheme even if the hash functions are replaced by another. This tricks is often adopted in many formal proofs of cryptographic functions.

Adversary's Goal

The goal of $\mathcal{A}$ is to generate any access key $K_{i,t}$ such that it is not covered by the union of her subscriptions. If $\mathcal{A}$ is able to generate such a key with non-negligible probability, we declarer that $\mathcal{A}$ is success in breaking the scheme.

Assumptions

We state the following assumptions that found the security of the scheme:

1) **Tamper resistant smart card.** Keys and algorithms stored in the smart card cannot be hacked or alternated by the $\mathcal{A}$.
2) **Unforgeable signature**. This states that $\mathcal{A}$ cannot produce any valid message-signature pair with probability larger than δ if they are not generated by the vendor. Notice that δ is a negligible function of τ, that is, for every positive integer c there exists an $N_c > 0$ such that for all $\tau > N_c$,
$$0 \leq \delta(\tau) < \frac{1}{\tau^c} \tag{9}$$
3) **Strong hash function**. It is difficult to find any pair of strings that have the same message digest by the hash function H.
4) **Secure HMAC**. It is difficult to compute $H_K(M)$ if only K is unknown. We assume that the key K has a length of k_1 bit and k_1 is a function of τ. We assume that the output length of the HMAC function is k_2-bit and k_2 is a function of τ.

CONSTRUCTION OF SIMULATOR

We construct a simulator interacting with $\mathcal{A}$. We will show that if $\mathcal{A}$ can break out protocol with non-neglibile probability, she will generate a valid message-signature pair with non-neglibile probability. Suppose $\mathcal{A}$ making k subscriptions $\mathbb{S} = \{S_1, S_2, \ldots S_k\}$. In each subscription S_i, we denote that $\mathcal{A}$ subscribes to C_{κ_i} for the time interval $[t_{i,1}, t_{i,2}]$, where $1 \leq \kappa_i \leq w$ and $t_{i,1} < t_{i,2}$. $\mathcal{A}$ is given a smart card π_i and K_{κ_i} for each subscription S_i, where K_{κ_i} is the class key for class C_{κ_i}. The value of K_{κ_i} are generated as specified by the protocol. According to the protocol, the simulator also computes $R_{a,b}$ for every class $C_b \prec C_a$. In addition, the simulator also outputs the signatures for every K_a and $R_{a,b}$.

$\mathcal{A}$ can input legitimate class key K_{κ_i} and $R_{\kappa_i,j}$ together with their signatures to the smart card π_i to obtain $K_{j,t}$, where $t_{i,1} \leq t \leq t_{i,2}$. The simulator will compute the following:

$$D = ((K_j)_Y \oplus H^t(a) \oplus H^{Z-t}(b) \oplus ID_j).$$

The simulator computes $\mathcal{H}_2(D)$ and records the pair $\{D, \mathcal{H}_2(D)\}$ into a private list. $\mathcal{H}_2(D)$ will be replied to $\mathcal{A}$ as the output of π_i.

Each smart card π_i has specified its subscription class C_{κ_i} and interval $[t_{i,1}, t_{i,2}]$. If $t' \notin [t_{i,1}, t_{i,2}]$, π_i should be unable to compute $H^{t'}(a)$ and $H^{Z-t'}(b)$. Therefore, the simulator will reject the request for computing $K_{j,t'}$ on behalf of π_i if $t' \notin [t_{i,1}, t_{i,2}]$. In addition, since only signed $R_{a,b}$ can be inputted into π_i, π_i will only compute for class j if: 1) $C_j \prec C_{\kappa_i}$ (signed by the simulator), or 2) $R_{\kappa_i,j}$ is signed by $\mathcal{A}$. If case 2) does not happen, both the class factor and the time factor are restricted, π_i will only compute the keys for the subscription S_i.

π_i will generate the key $K_{j,t}$ only if case 2 happens: $\mathcal{A}$ somehow computes a legitimate $R_{\kappa_i,j}$ such that $C_j \not\prec C_{\kappa_i}$ (that might involve solving discrete log problem) and creates the signature of it. However, this will only happens with probability less than δ.

$\mathcal{A}$ can query $\mathcal{H}_1$ and $\mathcal{H}_2$. When each query is made, the simulator will record its the input and output in its private list.

Let N be the event $\mathcal{A}$ outputs a key that is out of its subscription period, let say $K_{j,t'}$ and let η be the probability associated with this event. We denote the event E be the case that $\mathcal{A}$ has queries $H_K((K_j)_Y \oplus H^{t'}(a) \oplus H^{Z-t'}(b) \oplus ID_j)$ from oracle $\mathcal{H}_2$.

We can immediately derive the following equation from definitions:

$$\eta = \Pr(N|E)\Pr(E) + \Pr(N|\bar{E})\Pr(\bar{E}) \tag{10}$$

We have already argued that the key $K_{j,t'}$ is not generated by any π_i if $\mathcal{A}$ cannot forge a signature. Also, there is no other query from $\mathcal{H}_2$ in the scheme except the decryption phase by the smart card. $\mathcal{A}$ can only guess the output of the HMAC when if E does not happen, thus

$$\Pr(N|\bar{E}) = 2^{-k_2} + \delta \tag{11}$$

In addition, $\mathcal{A}$ does not have the key K, therefore, the probability for event E happening is bounded by finding K from HMAC, that is

$$\Pr(E) \leq q_H \times 2^{-k_1}, \tag{12}$$

where q_H is the total number of $\mathcal{H}_2$ queries made by $\mathcal{A}$.

By combining (10), (11), (12), we can conclude that:

$$\eta \leq 2^{-k_2} + \delta + 2^{-k_1} q_H, \tag{13}$$

If η is non-negligible, δ, the probability for $\mathcal{A}$ forging a signature, will also be non-negligible, that violates the security assumption. Therefore, η must be negligible for sufficient large τ.

□