An efficient key-management scheme for hierarchical access control based on elliptic curve cryptosystem

Fun-Gwo Jeng, Chung-Ming Wang
The Journal of Systems and Software 2006

Speaker: Wei-Lun Huang
Date: 2009/08/05
Outline

- Introduction
- A partially ordered user hierarchy
- An efficient key-management scheme for hierarchical access control based on elliptic curve cryptosystem
  - Key generation algorithm
  - Key derivation algorithm
  - Example of adding and deleting a class
- Conclusion
Introduction

- In an organization, the users and their authorized data are organized into a group of disjoint sets of security classes, and each user is assigned to a certain security class called his security clearance.

- In this scheme, each class in the hierarchy is allowed to select its own secret key. And a central authority (CA) is requires.

- The problem of efficiently adding or deleting classes can be solved without the necessity of regenerating keys for all the users in the hierarchy.
A partially ordered user hierarchy

Fig 1:
Key generation algorithm (1/3)

CA:

- Step 1: CA determines an elliptic group $E_p(a,b)$, and selects a base point $G = (x, y)$ from $E_p(a,b)$, and make $G$ public.

- Step 2: CA selects an algorithm $\tilde{A}:(x, y) \rightarrow \nu$, for representing a point on $E_p$ as a real number $\nu$, and make $\tilde{A}$ public.

- Step 3: CA choose a secret parameter $n_{CA}$, and make $P_{CA}$ public, where $P_{CA} = n_{CA} \cdot G$. 
Key generation algorithm(2/3)

- Every classes
  - Step 1: Each $C_i$ choose its own secret key $K_i$ and secret parameter $n_i$.
  - Step 2: Makes the $P_i (= n_i \cdot G)$ public.
  - Step 3: Encrypts the point $(K_i, n_i)$ by adding $kP_{CA}$ to it, and send point $\{kG, (K_i, n_i) + kP_{CA}\}$ to CA, where $k$ is a positive integer selected randomly.
Key generation algorithm (3/3)

- CA
  - Step 1: When CA got \{kG, (K_i, n_i) + kP_{CA}\} from every \(C_i\), CA will use \(n_{CA}\) to decrypts and derive \((K_i, n_i)\).
    \[(K_i, n_i) + kP_{CA} - n_{CA}(k \cdot G) = (K_i, n_i) + k(n_{CA} \cdot G) - n_{CA}(k \cdot G) = (K_i, n_i)\]
  - Step 2: CA construct the polynomials \(H_i(x)\) for every \(C_i\).
    \[
    H_i(x) = \prod_i (x - \tilde{A}(n_i, P_i)) + K_i
    \]
Key derivation algorithm (1/4)

Each class

- Step 1: Get the public parameters $H_j(x)$ and $P_j$ of $u_j$.

- Step 2: Computer $H_j(\tilde{A}(n_i P_j))$ and $K_j$ can be obtained.
Key derivation algorithm (2/4)

- Example: If \( C_i \) wants derives \( K_j \) from \( C_j \) (\( C_j \leq C_i \)).
  - Step 1: \( C_i \) have \( H_j(x) \) and \( P_j \) of \( C_j \).
    - \( H_j(x) = \prod (x - \tilde{A}(n_jP_t)) + K_j \)
  - Step 2: \( C_i \) use \( n_i \) to compute \( H_j(\tilde{A}(n_iP_j)) \).
    - \( H_j(\tilde{A}(n_iP_j)) = \prod_{C_j \leq C_i} (\tilde{A}(n_iP_j) - \tilde{A}(n_jP_t)) + K_j \)

\[
\begin{align*}
&= \{((\tilde{A}(n_iP_j) - \tilde{A}(n_jP_t)) \times \prod_{C_j \leq C_i, C_i \neq C_i} (\tilde{A}(n_iP_j) - \tilde{A}(n_jP_t))) + K_j \\
&= \{((\tilde{A}(n_iP_jG - \tilde{A}(n_jP_jG)) \times \prod_{C_j \leq C_i, C_i \neq C_i} (\tilde{A}(n_iP_j) - \tilde{A}(n_jP_t))) + K_j \\
&= K_j
\end{align*}
\]
Key derivation algorithm (3/4)

Example of Fig 1:

- $H_1(x) = \text{nil}$, which means that no other class has access to $C_1$.
- $H_2(x) = (x - \tilde{A}(n_2P_1)) + K_2$
- $H_3(x) = (x - \tilde{A}(n_3P_1)) + K_3$
- $H_4(x) = (x - \tilde{A}(n_4P_1)) + K_4$
- $H_5(x) = (x - \tilde{A}(n_5P_1))(x - \tilde{A}(n_5P_2)) + K_5$
- $H_6(x) = (x - \tilde{A}(n_6P_1))(x - \tilde{A}(n_6P_2))(x - \tilde{A}(n_6P_3))$
  \[\times (x - \tilde{A}(n_6P_4)) + K_6\]
- $H_7(x) = (x - \tilde{A}(n_7P_1))(x - \tilde{A}(n_7P_4)) + K_7$
Key derivation algorithm (4/4)

Example: If \( C_1 \) wants derived \( K_4 \) from \( C_4 \).

Step 1: \( C_1 \) have \( H_4(x) \) and \( P_4 \) of \( C_4 \).

\[ H_4(x) = (x - \tilde{A}(n_4P_1)) + K_4 \]

Step 2: \( C_1 \) use \( n_1 \) to compute \( H_4(\tilde{A}(n_1P_4)) \).

\[ H_4(\tilde{A}(n_1P_4)) = (\tilde{A}(n_1P_4) - \tilde{A}(n_4P_1)) + K_4 \]

\[ = (\tilde{A}(n_1n_4G) - \tilde{A}(n_4n_1G)) + K_4 = K_4 \]
Example of adding and deleting a class

1. If a class adding to the hierarchy:
   - Let $C_8$ be added as in immediate successor of $C_4$.
     - Step 1: $C_8$ selects its own secret key $K_8$ and secret parameter $n_8$.
     - Step 2: Makes $P_8 = (n_8G)$ public, and sends CA \{kG, (K_8, n_8) + kP_{CA}\}.
     - Step 3: CA will generates $H_8(x)$.

2. If a class removed to the hierarchy:
   - Example by $C_j$:
     - Just only delete $K_j, n_j$ and $P_j$. 
Conclusion

- This scheme ensures that the collaboration of a group of users is unable to reveal their predecessor’s secret key.

- Similarly, unable to generate their sibling’s secret key.

- It is flexible on the key selection, since any security class is able to select its own secret key $K$ for its own convenience, and able to change its secret key from $K$ to $K'$ for some security reason.
行政工作

- 帳款整理
  - 國科會結帳