An Efficient & Secure key management scheme for dynamic hierarchical access control

MS Research Thesis

By

Muhammad Asif

Reg: # 570-FBAS/MSCS/F09

Supervised By

Shehzad Ashraf Chaudhry

Department of Computer Science and Software Engineering

Faculty of Basic & Applied Sciences

International Islamic University Islamabad

- Elliptic Curve Cryptosystem
- Security Key
- Software engineering.
- theses

Department of Computer Science & Software Engineering

International Islamic University Islamabad

Dated: 10/02/2015

Final Approval

It is certified that we have examined the thesis report submitted by Mr. **Muhammad Asif, Reg No: 570-FBAS/MSCS/F-09**, and it is our judgment that this research project is of sufficient standard to warrant its acceptance by the International Islamic University, Islamabad for the Degree of Master of Science in Computer Science.

Committee:

External Examiner

Dr. Muazzam A.Khan Khattak

Assistant Professor,

Department of Computer Engineering ,

College of E&ME, National University Science &Technology (NUST).

Internal Examiner

Professor Dr. Muhammad Sher

Chairman Department of CS & SE/Dean FBAS.

International Islamic University (IIU), Islamabad.

Supervisor

Mr. Shehzad Ashraf Chaudhry

Lecturer Department of CS & SE,

International Islamic University (IIU), Islamabad.

ABSTRACT

Hierarchical access control system in a user hierarchy is used to give the access to sensitive information for those users which are in the user hierarchy (authorized users). Authorize users and their own information can be managed into a number of disjoint SC's (security classes) in order to their responsibilities. Every SC security class in a hierarchy is given an encryption key and can derive the encryption keys of all the lower SC security classes in order to predefine partially ordered relation $SCi \geq SCj$. In Jeng-Wing scheme and also in the Chung et al's scheme proposed an efficient scheme on access control in a user hierarchy based on elliptic curve cryptosystem. Their scheme gives a solution of key management efficiently for dynamic access problems. But in there schemes, there is a possibility of an exterior root finding attack, under this attack an adversary (attacker) who is not a user in any security class SC in a user hierarchy attempt to derive the secret key of a security class by using root finding algorithm. When in a user hierarchy add/delete or updating a relationship between two security classes. There is a security leak in both schemes. In the Nikhooghadam et al.'s and Wu et al's schemes they also provide a secure key management scheme for access control in a user hierarchy based on ECC. But there schemes requires large storage space in a public and private domain and also require huge computation time to calculate the private keys. So the proposed scheme provide an efficient and secure scheme in which both the problem is addressed i.e. the proposed scheme is secure against the exterior root finding attack and also require low storage space in public and private domain and also require low computation time.

DECLERATION

It is hereby declared that this work, neither as a whole nor as a part, has been copied out from any source. It is further declared that I have conducted this research and have accomplished this thesis entirely on the basis of my personal efforts and under the sincere guidance of my supervisor Mr. Shehzad Ashraf Chaudhry. If any part of this thesis is proved to be copied out from any source or found to be reproduced from some other project, I shall stand by the consequences. No portion of the work presented in this dissertation has been submitted in support of any application for any other degree of this or any other university or institute of learning.

Muhammad Asif

570-FBAS/MSCS/F09

A Dissertation submitted to the

Department of Computer Science and Software Engineering

International Islamic University Islamabad

In partial fulfillment of the requirements

For the degree of

Master of Science in Computer Science

2014

DEDICATION

THIS THESIS

IS DEDICATED TO MY

FATHER WHO TAUGHT ME

THAT HOW DIFFICULT THE TASK MAY

BE, ALWAYS TRY TO SOLVE IT YOURSELF. IT IS

ALSO DEDICATED TO MY MOTHER, WHO

TAUGHT ME THAT EVEN THE LARGEST

TASK CAN BE ACCOMPLISHED

IF IT IS DONE ONE STEP

AT A TIME.

ACKNOWLEDGEMENTS

All praise to Almighty Allah Who has all the names, and Who needs no name the most generous, considerate, and compassionate Who has blessed mankind with this verdict to think, explore, to learn and discover the hidden secrets of this universe and helped me to broaden the veils of my thought and enabling me to get through the difficulties indulged during this project. Also admiration to our beloved **Prophet Muhammad (PBUH)** who is always a great source of inspiration of divine devotion and dedication to me.

I would cordially pay my special appreciations and whole heartedly considerations to my reverend supervisors **Mr. Shehzad Ashraf Chaudhry** for his endless support, guidance and coordination while conducting this project. I owe him a great respect and honor and I am privileged to work under their supervision. It was their efforts, courage, moral support and endeavoring attitude that helped me to get through any problem or difficulty during each step of this project.

I would also like to pay my gratitude to all my respected teachers making me capable of what I am today due to their guidance and help. Thanking **Prof Dr. Muhammad Sher, Assistant Professor Hasnain Naqvi, Mr. Tawab Khan** and **Mr. Muhammad Qasim Khan** for their views which helped me in improving my research, also Mr. Bilal Shah for providing the managerial and administrative support.

Finally, my beloved parents and family, who deserve the credit more than I could ever express for always being completely supportive to me. They have been a constant source of advice, love and devotion to me. From moral to financial they have been blessing me with all the support that I needed up till now in my life.

Muhammad Asif

570-FBAS/MSCS/F09

Table of Contents

List of Tables

List of Figures

Chapter 1
Introduction

1. INTRODUCTION:

In this section we have discuss about the hierarchical access control which can be used in different applications like computer science, database and schools etc. And also the problem of the hierarchical access control. In which disuses the access rights that which security class can access the information and which SC can't access the secret information. We have also discuss that how the problem arises, when and who worked on this area of access problem. In this part we also discuss the solutions of different author's. We have also discussed about the elliptic curve cryptography and ECC over real numbers, the algebraic and geometric description of addition. At the end describe the encryption and description of the elliptic curve and show the comparison of key sizes in terms of computational efforts for cryptanalysis.

1.1 Hierarchal access control

Hierarchical access control system is one of the essential problems in computer network system. The hierarchy is used in number of applications i.e. computer networks, database management systems, school, military and government. All users in such areas use the hierarchical access control system and assigned a many disjoint sets of classes are called the security classes. Each class has its own resources associated with them. Classes hierarchies are arises when one class accesses the resources of the lower class in the hierarchy, so the user of a specific class access the resources of its own and the resources of the sub-classes in the user hierarchy. The resources in the hierarchy are encrypted under some keys and can be accessed using the encryption keys means that the resources are secured with the keys. When the users join the class in the hierarchy then gives them some secret information that will allow him to access the resources of its own and its sub-classes in the hierarchy.

Due to efficiency, key is assigned to every class in the hierarchy and every user assign one or some small number of keys allows them to access the secret data without involvement of the server. Through the key derivation process, it's clear that a scheme allow low requirements to be employed in a larger range of devices and the applications for example (embedded processors, small battery operated sensors and low cost smartcards etc.) than the high cost schemes. So the goal of the key management scheme in the hierarchy is to give the keys to users and the resources for the achievement of the efficiency and authentication. There are some criteria for achieving the efficiency in the key management scheme which includes:

- Every user must store the number of secret keys.
- A user needs to perform the calculation to access to the desired resources.
- If the hierarchy is updated and degree to which the user's private keys are affected.
- The system maintains the size of the information.

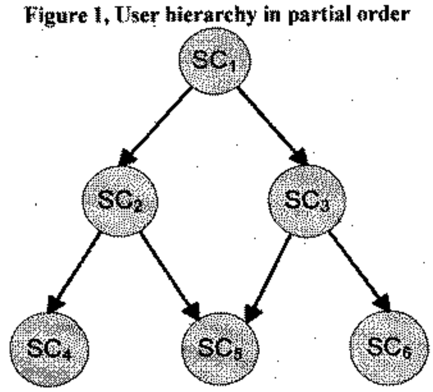
In the access control system security comes from their ability to refuse the access unauthorized information. E.g. in the fig (1) which shows the hierarchy of access control system in which the unauthorized users cannot able to access the resources of other classes than its sub-class.

Hierarchical access control is used in wide range of applications such as:

- Subscription of packages of pay TV, magazines and newspapers etc. Where packages are organized in a hierarchy that can be access by the users. For example a gold package will include everything in the package of silver package and also additional premium services.
- Role Base access control model, which is very useful for many organization to access the resources. Hierarchal access in these organizations is natural process and organized naturally. Where higher class can access its own resources and also its subordinates resources in the hierarchy.
- Cryptographic directories or file system, where user's access is based on the hierarchical relationship.
- Digital piles such as music and digital libraries, where different levels are access granted to the users.
- Hierarchal access control is used in operating systems (database and networking).

Hierarchal access is modeled as directed graph, where each node represents the class and edges represent the relation between the nodes. And the higher class in the hierarchy is called the predecessor and the lower class is called the successor. So the security class say $SC = \{SC1, SC2, SC3, \dots \dots SCn\}$ and the binary relation is partially order. In $(SC_i <=)$, $SC_j <= SC_i$ its means the security of SC_j is lower compare to SCi and SCi has a high security clearance then SCj as shown in the fig (1).

Figure 1, User hierarchy in partial order



In fig (1) the relation $SC_1 \longrightarrow SC_2$ shows the SC_1 have high security as compare to SC_2. The relationship shown in the hierarchy means the SC_i is the predecessor of SC_j and the SC_j is a successor of SC_i, in simple words the users in SC_i access the encrypted information in SC_j class and SC_j class has no right to access any data of SC_i. Each SC security class is give the private key Ki is used which will be used for encryption and decryption of the sensitive information. Figure shows an example user hierarchy which is partially ordered. If SCi would like to return information encoded by SCj class, it should have a correct key. Researchers proposed some schemes to solve the problem of access control in the hierarchy. In which first proposed a solution of the problem given by Akl and Taylor [1] that each SC security class is given a secret key and public parameter but the Akl and Taylor [1] there is some disadvantages in the scheme, which is the public data size will increase consistently as the security classes grows. The Mackinon et al.'s [2] scheme proposed an "optimal algorithm" which is known as the cannonical assignment that reduces public parameters importance but it is very hard to find an optical cannonical algorithm. The Harn and Lin [3] gave a key generating scheme using bottom-up approach, except using a top-down method as adopted as in the previous schemes, at any time when a new class is removed or added into the users hierarchy, the scheme discuss above cannot fulfill the requirements of the security, so the keys will be regenerated to issued all of them. To reduce the problem of dynamic access control, many schemes have been proposed.

The Chang et al., [4] presented key management scheme based on Newton's interpolation method and one-way function. In paper [4], a class with the high security clearance should do repeatedly the derivation steps to get the secret keys of the classes which are in the lower security classes. This process is an inefficient for the key derivation. The Wu-Cheng and Shen-

Chen [7] scheme, the access problem solved by using polynomial interpolation, the system did not require to continue the security classes and private keys of any user can easily update own private keys due to certain security reasons. So Hsu and Wu [11] find some security weaknesses in two schemes [11]. Attackers will violet the access control which is already defined, the policy to access the un-authorized data. Later Yang and Li [8] presented a scheme which is based on one way hash function the cryptographic key assignment scheme. The Hsu et al.'s [5] scheme printed out few security weaknesses in the Yang and Li paper [8] to declare that the claimed security requirement is breach i.e. the user able to over steps own right to get information which is not permitted. The Hsu at al.,[5] for than proposed improvement to exclude the highlighted flow in spite of that the Yang and Li[8] and Hsu et al.'s [5] not be able to performed the key updation an efficiently. Jang-Wang [6] scheme present an efficient key management and derivation scheme based on ECC. The Jang-Wang [6] presents the private key of every SC is decided by their own in place of trusted control authority. Main benefit of Jang and Wang [6] is the efficiently solve the dynamic key management scheme problem. That is not necessary to again produce keys for every SC's in the user hierarchy whenever the SC is added or deleted from the hierarchy. The compromising attack highlighted by Yu-Chein on Jang-Wang [6] scheme. Which clearly shows that their scheme does not achieves the required objective.

1.2 Elliptic Curve Cryptography

ECC is the public key encryption and decryption method which is much improves or efficient cryptographic used to make smaller and faster keys.

1.2.1 Elliptic Curve (EC) over Real Numbers

EC is not ellipses. These just like name because they are given by cubic equations; look like to these employs for computing the circumferences of an ellipse. In most, cubic equation for the EC takes the form

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

In which a,b,c,d and e belongs to real numbers also take value of x and y. For our use, it is enough to limit ourselves to the equations

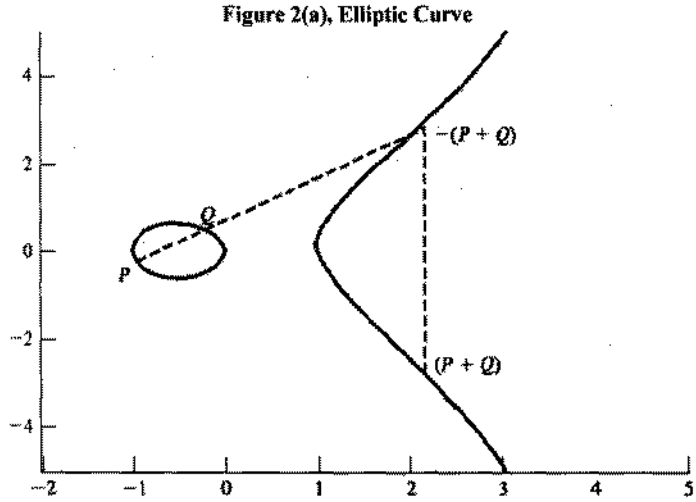
$$y^2 = x^3 + ax + b \tag{1.1}$$

These eq's are called cubic and degree three, because the largest exponent of the equation is three. They are also consider as in the explanation of an EC as a single element denoted as O and known as the zero point or point at infinity, to draw such kind of a curve, require to calculate

$$y = \sqrt{x^3 + ax + b} \qquad (1.2)$$

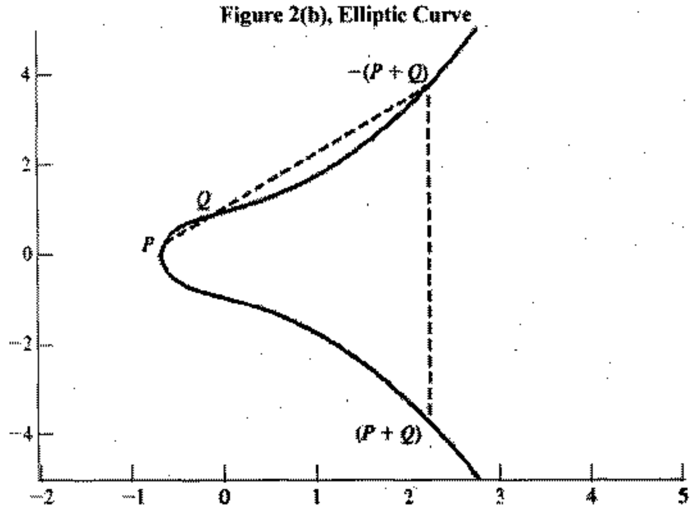
For a and b given values of, the plot made up of negative and positive values of y for the value of x. every curve is symmetric about $y = 0$. Figure 1.1 elliptic curves indicate two examples. As can easily be seen, sometimes the formula creates strange looking curves. Think about the $E(a, b)$ to be composed of all the points (x, y) which satisfy eq (1.1) both using the element O. using a two different values (a, b) of the results in a different set $E(a, b)$. Applying this method, both curves in fig 1.1 shows the $E(-1, 0)$ and $E(1, 1)$, respectively.

Figure 2(a), Elliptic Curve



a) $y^2 = x^3 - x$

Figure 2(b), Elliptic Curve



b) $y^2 = x^3 + x + 1$

1.2.2 Geometric Description of Addition

We know that a group can be defined based on the *E (a, b)* for some given values of *a* and *b* in eq (1.1).

$$4a^3 + 27b^2 \neq 0 \tag{1.3}$$

The definition of a group, first define the process known as addition and denoted as '+', for the set E (a, b) where *a* and *b* satisfy the eq (1.3). in geometric terms, the addition rules can be said as come after, if three points on an EC lie on a straight line, their sum is *O*. form the definition, the addition rules over an elliptic curve are clearly defined.

a) *O* performs the additive identity. So $O = -O$, for any point P on the EC, $P + O = P$. In which suppose $P \neq O$ & $Q \neq O$
b) The -ve P is the point having similar x coordinates and the -ve y coordinates, i.e. if $P = (x, y)$, then $-P = (x, -y)$. Impression that the two points able to combined by a vertical line. And $P + (-P) = P - P = O$.
c) *P* and *Q* added having different x coordinates; a straight line is drawn through these points then search the third point of intersection R i.e. the third point of intersection fig 2(a) shows this construction.

d) The geometric meaning to explain the next item also put P and $-P$, having similar x coordinates, the points are combined by a straight line, and can also be seen the dividing curve at the infinity point, so $P + (-P) = O$.

e) Double a point Q, draw the tangent line to search the other intersection point S so $Q + Q = 2Q = -S$

1.2.3 Algebraic Description of Addition

This section mentioned few outputs which make computation of addition over EC. Using two different points $P = (xP, yP)$ and $Q = (xQ, yQ)$ and not -ve of one another, the slope of line l that combines it as $\Delta = (yQ - yP)/(xQ - xP)$. Here is absolutely another point which l cuts the EC, can be express $R = P + Q$ as follows,

$$x_R = \Delta^2 - x_P - x_Q \tag{1.4}$$

$$y_R = -y_P + \Delta\,(x_P + x_R) \tag{1.5}$$

There is need to add a point to itself: $P + P = 2P = R$ where $y_P \neq 0$ and the expression are

$$x_R = (3x^2{}_P + a/2y_P) - 2x_P \tag{1.6}$$

$$y_R = (3x^2{}_P + x/2y_P)(x_P - x_R) - y_P \tag{1.7}$$

1.2.4 Elliptic Curve over Z_P

Elliptic curve cryptography makes use of EC that is the coefficients and variables are all limited to elements of finite field. In cryptographic application there are two families of the elliptic curves are employ, prime curve over Z_P and binary curve over $GF(2^m)$. the Z_P over prime curve, use cubic equation where the coefficients and variables all take the values in the set of integers between 0 to p − 1 so the computations are performed on modulo p. Binary curve define over $GF(2^m)$, the coefficient and the variables all takes on the values $GF(2^n)$ and then computations are done using $GF(2^n)$. To indicate that prime curve is good for s/w applications due to larger bit fiddling process required by the binary curve is not needed and the binary curve is good for the h/w applications, because it takes some logic gates to make a quick and powerful cryptosystem.

1.2.5 Elliptic Curve (EC) over GF (2^m)

The finite field $GF(2^m)$ is made up of 2^m elements and both addition and multiplication process may be defined over polynomials. The EC over $GF(2^m)$ use a cubic equation, where the variables and coefficients take on the values in GF(2^m) for few number m and computations are done using the arithmetic rules in GF(2^m). The cubic equation is helpful for the cryptographic applications for the EC is not similar for GF(2^m) than the Z_P

$$y^2 + xy = x^3 + ax + b \tag{1.8}$$

So it is very clear that the coefficient a and b and the variable x and y are the elements of GF(2^m) and the computations are performed in GF(2^m).

1.3 Encryption and Decryption using Elliptic Curve (EC)

There are several methods has been analyzed using elliptic curve. First take simple text message m to encrypt and sent to the x and y point Pm, Pm point will be encoded as secret text and then to decrypt, the message as coordinated x or y cannot be encrypted simply for some reason not all coordinates are in Eq (a, b).

Key exchange system the encryption and decryption needs a point G and elliptic group Eq (a, b) as points. Every user A chooses a secret key n_A and generates a public key

$$P_A = n_A \, x \, G. \tag{1.9}$$

To encode and send a message Pm to B, A selects a random +ve integer k and make the cipher text Cm made up of two points.

$$Cm = \{ kG, Pm + k P_B \}$$

Remember A has used B's public key P_B to decode the cipher text; B multiplies the first point in the pair by B's private key and subtracts the result from the 2nd point:

$$Pm = kP_B - n_B (k G) = Pm + k (n_B G) - n_B (k G) = Pm \tag{1.10}$$

A encode the message Pm by adding $kP_B$to it. No one except A aware of the value k and in fact P_B has a public key, and no one able to decode the kP_B . And the user A leave a clue that's enough to take off the mask if one can knew the secret key n_B. The adversary to get back the message and he has calculated the k given G and kG which is supposed hard.

1.3.1 Elliptic Curve Cryptosystem (ECC) Security

The EC security relies that how hard it is to decide k given kP. That is referring to logarithm problem of ECC. Pollard rho technique is known as the fastest technique of CC logarithm.

Table 1. Computational efforts for cryptanalysis in the terms of key sizes comparisons

Symmetric Scheme (key size in bits)	ECC Based scheme (size of n bits)	RSA and DSA (modulus size in bits)
56	112	512
80	160	1024
112	224	2048
128	256	3072
92	384	7680
256	512	15360

The table 1 shows the comparison of different algorithms showing the same kind of key size in terms of calculations effort for cryptanalysis. So it can show from the outputs a greater lesser key size may be employ for EEC compared to RSA. And also for same key lengths the calculated hard work need for RSA and ECC is comparable [JURI97]. So here are benefits of computation using elliptic curve cryptosystem having small key size as compare to RSA.

Chapter 2

Literature Review

2. Literature Review

In this section we have discussed about the hierarchical access control problem. Starting from the symmetric, RSA encryption algorithm and up-to the elliptic curve encryption and decryption method that how and which method is efficient and required smaller key. Researchers proposed some schemes to solve the problem of access control in the hierarchy. In which first proposed a solution of the problem given by Akl and Taylor [1] that each SC security class is given a secret key and public parameter but the Akl and Taylor [1] there is some disadvantages in the scheme, The Mackinon et al.'s [2] scheme proposed an "optimal algorithm" which is known as the cannonical assignment that reduces public parameters importance. The Harn and Lin [3] gave a key generating scheme using bottom-up approach, except using a top-down method as adopted as in the previous schemes, at any time when a new class is removed or added into the user's hierarchy. After that the Chang et al., [4] presented key management scheme based on Newton's interpolation method and one-way function. The Wu-Cheng and Shen-Chen [7] scheme, the access problem solved by using polynomial interpolation, the system did not require to continue the security classes and private keys of any user can easily update own private keys due to certain security reasons. So Hsu and Wu [11] find some security weaknesses in two schemes [11]. Later Yang and Li [8] presented a scheme which is based on one way hash function the cryptographic key assignment scheme. The Hsu et al.'s [5] scheme printed out few security weaknesses in the Yang and Li [8] to declare that the claimed security requirement is breach. The Hsu et al.'s[5], Yang and Li[8] and Hsu et al.'s [5] not be able to performed the key updation an efficiently. Jang-Wang [6] scheme present an efficient key management and derivation scheme based on ECC. The compromising attack highlighted by Yu-Chein on Jang-Wang [6] scheme. Which clearly shows that their scheme does not achieves the required objective.

2.1 Review of Jeng -Wing scheme:

The Jeng-Wang [6] presented an efficient key management and derivation scheme based on ECC. For the solution of access control in the hierarchy. The Jing-Wing [6] scheme consist of three phases, initialization phase, key generation phase, and last the key derivation phase. In the first phase of initialization, central authority decides whole system parameters. In the 2nd phase, every security class SC decides a private key on an elliptic group over a finite field. All private keys are sent to CA through some safe path to build a key relationship derive from the hierarchy. In 3rd key derivation phase, the SC_i (predecessor) may employ his public information and private key which is connected to the SC_j (successor) to derive the encryption or decryption keys to access the authorized data.

2.1.1 1st key initialization phase:

Central Authority (CA) chooses a prime p number which will be large and an EC, $E_p(a, b): y^2 = x^3 + ax + b \bmod p$ with O at infinity, in which a and $b \in Z^*_p$ are two numbers which is random integer always satisfying that $4a^3 + 27b^2 \bmod p$ will not be equal to 0. Let $G \in E_p(a, b)$ is a base point having order q, and q is large prime number. CA (central authority) also chooses a transformation function $\tilde{A}: (x, y) \rightarrow v$ to transforms a point on $E_p(a, b)$ and $v \in Z^*_p$ at last the central authority shows publically the $(p, q, \tilde{A}, E_p(a, b), G)$.

2.1.2 2nd Key generation phase:

First central authority decides the secret points n_{ca} randomly then announce publically the $P_{ca} = n_{ca}G$. Ever class makes its private information as a (K, n) point. Then each security class selects randomly k which is +ve integer then produce, $\{kG, (K, n), + k P_{ca}\}$, Whenever CA obtain them, It will multiples n_{ca} which is private parameter then from the second point subtracts the result for the derivation of the (K,n):

$$(K, n) + k P_{ca} - n_{ca}(kG) = (K, n) + k(n_{ca} G) - n_{ca}(kG) = (K, n)$$

2.1.3 3rd Key derivation phase:

To derive (Ki, ni):

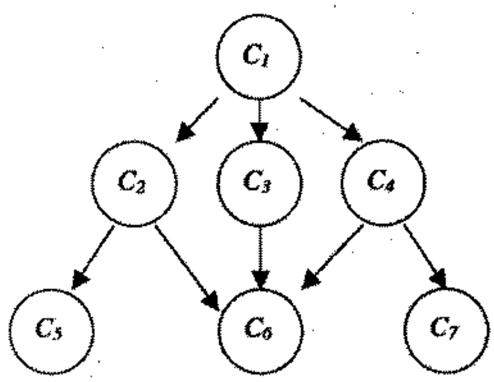
$(Ki, ni) + kP_{ca} - n_{ca}(kG)$

$= (Ki, ni) + k(n_{ca}G) - n_{ca}(kG) = (Ki, ni)$

For security class SCi 1<= i <= m, CA construct the polynomial Hi(x)

$Hi(x) = \Pi(x - \tilde{A}(niPt) + Ki \quad \text{for all } Ci < Ct$

Figure 3, User hierarchy partially ordered



An example using the hierarchy as shown in the fig 3. C_1 class decides his key K_1 which is private, and n_1 (secret parameter) and then generate the $P_1 = n_1G$ i.e. public parameter and then send these parameter to central authority. All the SC's in the user hierarchy will perform the similar procedure. Now the central authority gets every secret parameter and private keys to constructs the polynomial for every security class after that declare these points publically.

$H1(x) = nil$, *it means that no one can accesses the class C1*

$H2(x) = (x - \tilde{A}(n_2P_1)) + K_2$

$H3(x) = (x - \tilde{A}(n_3P_1)) + K_3$

$H4(x) = (x - \tilde{A}(n_4P_1)) + K_4$

$H5(x) = (x - \tilde{A}(n_5P_1))(x - \tilde{A}(n_5P_2)) + K_5$

$H6(x) = (x - \tilde{A}(n_6P_1))(x - \tilde{A}(n_6P_2))(x - \tilde{A}(n_6P_3))(x - \tilde{A}(n_6P_4)) + K_6$

$H7(x) = (x - \tilde{A}(n_7P_1))(x - \tilde{A}(n_7P_4)) + K_7$

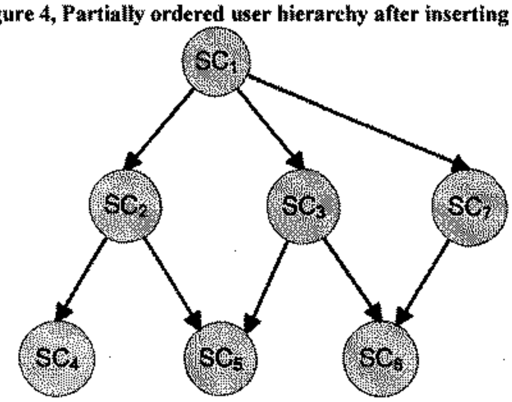
For example C4 as shown in the fig 3 interested to get the private key K_7 by using its own private n4 and public parameter H7(x) and P_7, it will be easily calculated the K_7 which is secret key shown below.

$$
\begin{aligned}
H7(x) &= (x - \tilde{A}\,(n_7P_4))\,(x - \tilde{A}\,(n_7P_1)) + K_7 \\
&= (\tilde{A}\,(n_4P_7) - \tilde{A}\,(n_7P_4))\,(\tilde{A}\,(n_4P_7) - \tilde{A}\,(n_7P_1)) + K_7 \\
&= (\tilde{A}\,(n_4\,n_7G) - \tilde{A}\,(n_7P_4))\,(\tilde{A}\,(n_4\,n_7G) - \tilde{A}\,(n_7P_1)) + K_7 \\
&= (\tilde{A}\,(n_7P_4) - \tilde{A}\,(n_7P_4))\,(\tilde{A}\,(n_4P_7) - \tilde{A}\,(n_7P_1)) + K_7 \\
&= K_7
\end{aligned}
$$

2.1.4 The Compromising Attack on Jeng-Weng Scheme:

On Jeng-Weng [6] scheme the compromising attack to declare that any user which is not belongs to the same hierarchy will not have knowledge to access unauthorized encrypted key, suppose that the relationship between any two SC's is changed. In Jeng-Weng [6], every SC produces his encrypted key as shown in the above equations. Which is employed to construct the polynomial $fj(x)$ for its successor SC_j where $SC_i >= SC_j$. Now CA can add and remove some predecessor into or from the SCj, central authority will update the polynomial as $f'_j(x)$ that is public.

Figure 4, Partially ordered user hierarchy after inserting SC7



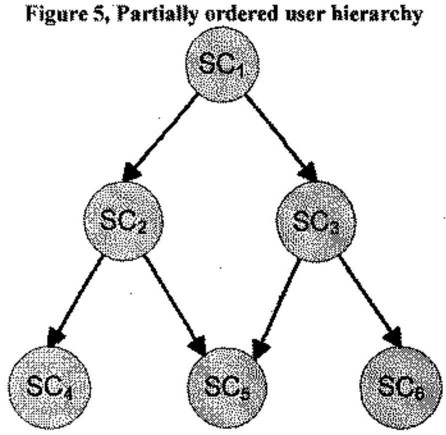
Let G_j is the set of the security classes SC_l's ($SC_i < SC_l$), and remains the predecessors of SC_j. $SC_l \in Gj$ is also give the new polynomial $f'_j(x)$. So its happen by the point $(v_{l,j}, k_{j,2})$ connected to the $SC_l \in G_j$ will satisfy $\Psi(vl,j) = 0$, where $\Psi(x) = fj(x) - f'j(x)$. Having information of $fj(x)$ and $f'_j(x)$, enemy able to attempt and derive all $v_{l,j}$'s and $\Psi(v_{l,j}) = 0$ using finding the roots of the polynomial $\Psi(x) = 0$, in the polynomial time[12]. Then a user which is not from the same hierarchy is not having any private data can easily proceed to get key $k_{j,2}$ which is encrypted of the security class SC_j by $k_{j,2} = f_j(v_{l,j})$. Hence Jeng-Weng [6] scheme is not secured. Example is to show that the attack, which discussed above is effective on Jeng-Weng [6] scheme. For example, hierarchy has seven SC's as in the fig 4. The predecessor of the security class SC_6 are SC_1, SC_3, and SC_7, and the public polynomial $f_6(x)$ for SC_6 is constructed by the points $(v_{1,6}, k_{6,2})$, $(v_{3,6}, k_{6,2})$ and $(v_{7,6}, k_{6,2})$. If the SC_7 is deleted from the hierarchy as shown in the fig 4, $f_6(x)$ the public polynomial for SC_6 will be changed with $f'_6(x)$ which is made up of the points $(v_{1,6}, k_{6,2})$ and $(v_{3,6}, k_{6,2})$ having the information of $f_6(x)$ and $f'_6(x)$, then anyone who is not belong to the hierarchy (attacker) can derive $v_{1,6}$ or $v_{3,6}$ by finding the roots of the polynomial $\Psi(x) = f_6(x) - f'6(x)$ in the polynomial time [12]. The adversary can proceed to use $v_{1,6}$ and $v_{3,6}$ to derive the key SC_6 by $k_{6,2} = f_6(v_{1,6})$ and $k_{3,6} = f(v_{3,6})$ by himself.

From the above analysis, few keys which are secret will be known to everyone if the public polynomial is changed and its few points are not changed. To take different situation in which public and secret keys of few predecessors of SCj modified. Central authority should change the public polynomial as $f'(x)$. Suppose G_j be the set of the security classes SC_l's $(SC_j < SC_l)$ where private and public keys are remain same. It may seen the $SC_l \in G_j$ are also gave the new polynomial $f'(x)$. So, any user which is belong to the hierarchy still able to derive all $v_{l,j}$'s for $SC_l \in G_j$ such that $(v_{l,j}) = 0$ where $\Psi(x) = f(x) - f'(x)$ [12] and then get the encryption key of SC_j as $k_{j,2} = f_j(v_{l,j})$. So the Jeng-Weng [6] scheme is not protected against compromising attack no matter whether the relation of any two SC's is not changed. Remember the similar example as in the fig (4) above that the predecessors of SC_6 are SC_1, SC_3 and SC_7, and the public polynomial $f_6(x)$ is constructed using the points $(v_{1,6}, k_{6,2})$, $(v_{3,6}, k_{6,2})$ and $(v_{7,6}, k_{6,2})$. Suppose the public and secret keys of SC_7 are modified $(k'_{7,1}, Y'_7 = k'_{7,1} G)$, the public polynomial $f_6(x)$ for SC6 will changed with $f'_6(x)$ that obtained using points $(v_{1,6}, k_{6,2})$, $(v_{3,6}, k_{6,2})$ and $(v_{7,6}, k_{6,2})$ where $v'_{7,6} = \tilde{A}(k_{6,1}\ Y'_7)$. Having the information of $f_6(x)$ and $f'_6(x)$, any outsider can access the $v_{1,6}$ or $v_{3,6}$ by finding the roots of the polynomial [12] $\Psi(x) = f_6(x) - f'_6(x)$. The SC_6 encryption key will be compromised by $k_{6,2} = f_6(v_{1,6})$ or $k_{6,2} = (v_{3,6})$.

2.2 Review of Chung et al scheme.

Chung et al.'s [9] contains the 1st relationship building phase, 2nd key generation phase and 3rd key derivation phase. In the 1st phase, a central authority constructs hierarchal model to control the access in order to the relations between the users. In 2nd phase, Central authority selects a two points and makes the public polynomial using one way hash function. Using 3rd phase, the predecessor in the user hierarchy may employ public and secret key data connected to the successor can gets the decrypted key of the successor for accessing the protected files,

Figure 5, Partially ordered user hierarchy



2.2.1 The relationship building phase:

The first phase, central authority constructs the hierarchal model for controlling the access in order to the relations means the users in the hierarchy. Let $U = (SC_1, SC_2, SC_3, \ldots, SC_n)$ set of n SC's in the hierarchy. Suppose that SCi having higher clearance then SC_j i.e. $SCi >= SCj$. A relation $(SCi, SCj) \in R_{ij}$ between pair of classes SC_i and SC_j exists in the hierarchy if SCi can access SC_j.

2.2.2 Key generation phase:

the 2nd phase, CA chooses high prime p randomly, the EC, $E_p\ (a, b)$ defined over Z_p such that the order of $E_p\ (a, b)$ lies between $[p + 1 - 2\sqrt{p}, p + 1 + 2\sqrt{p}]$, and $h(.)$ one way hash function to transfer a point to number and a base point G_j from $E_p (a, b)\ 1 <= j <= n$. So $SC_j\ 1 <= j <= n$, decides private key sk_j and a sub-private key s_j, for whole $\{SC_i / SC_i, SC_j \varepsilon R_{ij}\}$, to calculate $s_i G_j = (x_{j,i} // y_{j,i})$

and //is a bit concatenation operator. At last to calculate the public polynomial $f_j(x)$ using the values of $h(x_{j,i}//y_{j,i})$ as

$$f_j(x) = \prod_{sc_i > sc_j} (x - h(x_{j,i}//y_{j,i})) + sk_j \bmod p$$

Sends sk_j and s_j to the SC_j via secure path, and announces $p, h(), G_j, f_j(x)$ publically.

2.2.3 Key derivation phase:

The predecessor SC_i can compute all the successor (SC_j) keys of sk_j where $(SC_i, SC_j) \in R_{i,j}$ between the SC_i and SC_j hold shown below:

$$f_j(x) = \prod (x - h(x_{j,i}//y_{j,i}) + sk_j \bmod p$$

$$f_j(h(x_{j,i}//y_{j,i}) = sk_j \bmod p$$

E.g. user hierarchy in the fig 5, if C_1 wants to derive his successor C_4 encrypted information by deriving C_4 secret key sk_4.

$$f_4(x) = (x - h(x_{4,2}//y_{4,2}) (x - h(x_{4,1}//y_{4,1}))$$

$$s_iG_j = (x_{j,i}, y_{j,i})$$
$$C_1 = s_1G_4 = (x_{4,1}, y_{4,1})$$
$$h(x_{4,1}//y_{4,1})$$

$$f_4(h(x_{4,1}//y_{4,1})) = ((h(x4,1//y4,1) - h(x4,2//y4,2) * (h(x4,1//y4,1) h(x4,1//y4,1)) + sk_4 \bmod p$$

$$f4(h(x_{4,1}//y_{4,1})) = sk_4 \bmod p$$

2.2.4 On Chung et al's exterior root finding attack:

Existing now an exterior root finding attack on Chung et al.'s [9] paper, it proves his scheme is not secure. In the exterior root finding attack, the attacker try gets the private key of a SC through the root finding algorithm [12]. Note that in key generation algorithm [9] every SC SC_i, central authority produces the base point G_i, the sub-private key s_i and private key sk_i, decides public polynomial $f_j(x)$ after that safely sends ski to SC_i, and publicly announces p, $h(.)$, G_i and $f_i(x)$ of SC_i, the sub-private points of its all predecessors are fixed in its public polynomial $f_i(x)$.

Think about a situation in which SC_k added into hierarchy having relation $SC_i \geq SC_k \geq SC_j$. After SCk included as a predecessor of SC_j, central authority updates the public polynomial of SC_j by replacing $f_j(x)$ into $f'_j(x)$. Means that those predecessors which stayed as a predecessor of SC_j in $f'_j(x)$, now having the information of $f_j(x)$ of SC_j earlier included the SC_k and public polynomial $f'_j(x)$ of SC_j after inserting the SC_k when the private key sk_j of SC_j had been substitute by central authority, an adversary able to build a polynomial by taking the difference of $f_j(x)$ and $f'_j(x)$. And the difference is denoted as $\phi(x) = f_j(x) - f'_j(x)$. So

$$\phi(x) = f_j(x) - f'_j(x) = \Big(\prod_{SC_i > SC_j} (x - h(x_{j,i} // y_{j,i}) + sk_j \ \underline{mod p}\Big) - \Big(\prod_{SC_i > SC_k > SC_j} [(x - h(x_{j,i} // y_{j,i})][\,[(x - h(\underline{x_{j,k}} // \underline{y_{j,k}})] + sk_j \ \underline{mod p}\Big)$$

$$= \prod_{SC_i > SC_j} [(x - h(x_{j,i} // y_{j,i})] - \prod_{SC_i > SC_k > SC_j} [(x - h(x_{j,i} // y_{j,i})][\,[(x - h(x_{j,k} // y_{j,k})] + mod p$$

Further noticed the constructed $\phi(x)$ polynomial have something which is common $(x - h(x_{j,i} // y_{j,i})$, and adversary finds the roots of the equation $\phi(x) = f_j(x) - f'_j(x) = 0$ in a polynomial time [12]. Knowing the roots, the adversary easily derives the private key sk_j of SC_j. Gets the roots of $h(x_{j,i} // y_{j,i})$ after that calculates the private key sk_j of SC_j as $sk_j = f_j(h(x_{j,i} // y_{j,i})) = f'_j(h(x_{j,i} // y_{j,i}))$ $mod\ p$. It easily seen the Chung et al.'s [9] having some problems against exterior finding attack. Let's take the example to prove that on Chung et al.'s [9] exterior root finding attack is possible. The user hierarchy has six SC as in fig 5, mean $U = \{SC_1, SC_2, SC_3, SC_4, SC_5, SC_6\}$. CA calculates the EC polynomial $f_j(x)$ for every SC_j. Every SCi after that gets the private keys of his successors SC_j, through the key generation algorithm as below:

$$f_j(x) = \prod_{SC_i > SC_j} (x - h(x_{j,i} // y_{j,i}) + sk_j \bmod p$$

SC_1: $f_1(x) = [x - h(x_{1,0} // y_{1,0})] + sk_1 \bmod p$ where s_0 is given by CA

SC_2: $f_2(x) = [x - h(x_{2,1} // y_{2,1})] + sk_2 \bmod p$

SC_3: $f_3(x) = [x - h(x_{3,1} // y_{3,1})] + sk_3 \bmod p$

SC_4: $f_4(x) = [x - h(x_{4,1} // y_{4,1})]\ [x - h(x_{4,2} // y_{4,2})] + sk_4 \bmod p$

SC_5: $f_5(x) = [x - h(x_{5,1} // y_{5,1})]\ [x - h(x_{5,2} // y_{5,2})]\ [x - h(x_{5,3} // y_{5,3})] + sk_5 \bmod p$

SC_6: $f_6(x) = [x - h(x_{6,1} // y_{6,1})]\ [x - h(x_{6,3} // y_{6,3})] + sk_6 \bmod p$

Consider a situation when a new SC_7 is added in the hierarchy shown in the *fig 2(c)*, with the relation $SC_1 > SC_7 > SC_6$. The updated user hierarchy as shown in the *fig 2(d)*. Once SC_7 *is* inserted, central authority require to choose randomly sk_7, s_7 and G_7. Since SC_7 is a successor of SC_1 and a predecessor of SC_6, CA build the $f_7(x)$ that is public polynomial and replaces the $f_6(x)$ with $f'_6(x)$, and noted before that connects SC_7 into the hierarchy, so the polynomial of the elliptic curve for SC_6 was

$$f_6(x) = [x - h(x_{6,1} // y_{6,1})]\ [x - h(x_{6,3} // y_{6,3})] + sk_6 \bmod p \quad (1)$$

After combining the SC_7, the polynomial $f'_6(x)$ for SC_6 and $f_7(x)$ are formed as follows:

$$f'_6(x) = [x - h(x_{6,1} // y_{6,1})]\ [x - h(x_{6,3} // y_{6,3})]\ [x - h(x_{6,7} // y_{6,7})] + sk_6 \bmod p \quad (2)$$

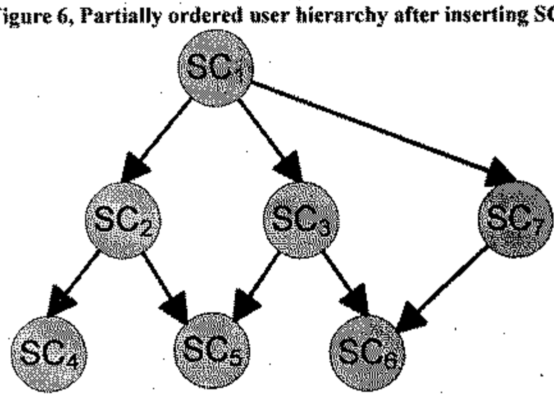
$$f_7(x) = [x - h(x_{7,1} // y_{7,1})] + sk_7 \bmod p \quad (3)$$

Now having the information of the polynomial $f_6(x)$ and $f'_6(x)$ in eqs (1) and (2), the adversary finds out the roots of the equation easily.

$$\phi(x) = f_6(x) - f'_6(x) = 0 \Rightarrow [x - h(x_{6,1} // y_{6,1})]\ [x - h(x_{6,3} // y_{6,3})]\ [1 - (x - h(x_{6,7} // y_{6,7}))] = 0 \bmod p \quad (4)$$

Solving the eq. (4), the adversary get the roots as x $x = h(x_{6,1} // y_{6,1})$, $h(x_{6,3} // y_{6,3})$ and $1 + h(x_{6,7} // y_{6,7})$. After finding these roots, $h(x_{6,1} // y_{6,1})$ *and* $h(x_{6,3} // y_{6,3})$ proves the eq 1 and 2. So having the knowledge of these values, the attackers without any difficulty calculates the sk_6 which is the secret key SC_6 as:

Figure 6, Partially ordered user hierarchy after inserting SC7



$SC = \{SC_1, SC_2, \ldots\ldots, SC_{n'}\}$

$$
\begin{aligned}
sk_6 &= f_6\,(h(x_{6,1}\,//y_{6,1}))\ (mod\ p) \\
&= f'_6\,(h(x_{6,1}\,//y_{6,1}))\ (mod\ p) \\
&= f_6\,(h(x_{6,3}\,//y_{6,3}))\ (mod\ p) \\
&= f'_6\,(h(x_{6,3}\,//y_{6,3}))\ (mod\ p)
\end{aligned}
$$

2.3 Review of the Nikooghadam, et al.

In this scheme [10], CA builds the a hierarchy model for get control in order to relationships between the SC's and then proposed having N SC's and they make a set $SC = \{SC_1, SC_2, \ldots\ldots, SC_{n'}\}$ SCi is greater security then SC_j, so the SC_j can be easily accessed by SC_i and this relation is represented as $SC_i >= SC_j$. In this scheme [10] there are two phases, key generation and key derivation phase, which discussed below.

2.3.1 Key generation phase:

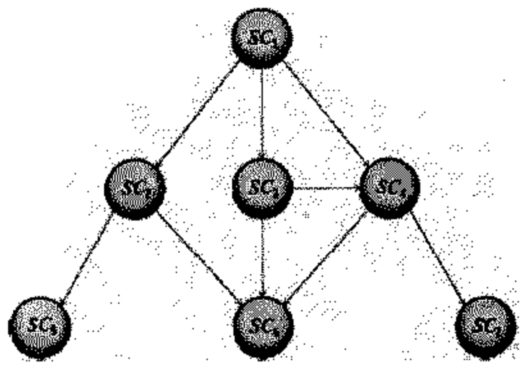
Central authority decide q of finite field Fq, where $q = p$ is an odd prime, or $q = 2^m$ and q is prime power. CA specify an appropriate elliptic curve by choosing the parameters for the coefficient a and b of EC equation E over $F_q : y^2 + xy = x^3 + ax^2 + b$. Then CA determine the G base point that's finite point on EC with the highest order n so $nG = O$. Then declare them publically by the CA values of E_q, G and n. The $f(x)$ hash function is picked by CA to transfer a point on EC E_q into a number v that $v \in F_q$ and makes the $f(x)$ public. Each security class SC_i selects the random integer d_i between $[1, n - 1]$ as its private parameter, after that calculates the

$P_i = d_iG$ and declares publically. CA selects a random integer ki between the $[1, n - 1]$ for every SC_i and calculates $Z_i = k_iG$. The private key of every SC is $Sk_i = f(Z_i)$. Each SC_j $1 <= j <= N$. $\forall$ SC_i which satisfies $SC_j <= SC_i$, central authority decides $M_{i,j} = k_j(P_i)$ and publically declare them.

2.3.2 Phase key derivation phase:

The relationship $SC_i >= SC_j$, SC_i decides private keys for whole his successors (SK_j) and his private key (SK_i). Determines d_i^{-1} and computes the $Z_j = k_jG = d_i^{-1} M_{i,j}$, and the d_i^{-1} shows the inverse of finite field, which is the EC digital signature algorithm *(ANSI, 1998)* one of the required operations. Determines the $SK_j = f(Z_j)$

Figure 7, partially ordered user hierarchy



In the fig (7) contains seven SC's denoted as $SC = \{SC1, SC2, SC3, SC4, SC5, SC6, SC7\}$. Central authority decides Mi,j public parameters then declare publically. Set of public parameters which are used to derive the private key of every SC through his predecessor.

$SC_1 = \{M_{1,1} = k_1(P_1)\}$

$SC_2 = \{M_{1,2} = k_2(P_1), M_{2,2} = k_2(P_2)\}$

$SC_3 = \{ M_{1,3} = k_3(P_1), M_{3,3} = k_3(P_3)\}$

$SC_4 = \{ M_{1,4} = k_4(P_1), M_{3,4} = k_4(P_3), M_{4,4} = k_4(P_4)\}$

$SC_5 = \{ M_{1,5} = k_5(P_1), M_{2,5} = k_5(P_2), M_{5,5} = k_5(P_5)\}$

$SC_6 = \{ M_{1,6} = k_6(P_1), M_{2,6} = k_6(P_2), M_{3,6} = k_6(P_3), M_{4,6} = k_6(P_4), M_{6,6} = k_6(P_6)\}$

$SC_7 = \{ M_{1,7} = k_7(P_1), M_{3,7} = k_7(P_3), M_{4,7} = k_7(P_4), M_{7,7} = k_7(P_7)\}$

So every SC able to gets his security key and private keys of its successors, shown below:

$SK_1 = f(Z_1) = f(d_1^{-1} M_{1,1})$ *so there is no predecessor of* SC_1

$SK_2 = f(Z_2) = f(d_1^{-1} M_{1,2}) = f(d_2^{-1} M_{2,2})$

$SK_3 = f(Z_3) = f(d_1^{-1} M_{1,3}) = f(d_3^{-1} M_{3,3})$

$SK_4 = f(Z_4) = f(d_1^{-1} M_{1,4}) = f(d_3^{-1} M_{3,4}) = f(d_4^{-1} M_{4,4})$

$SK_5 = f(Z_5) = f(d_1^{-1} M_{1,5}) = f(d_2^{-1} M_{2,5}) = f(d_5^{-1} M_{5,5})$

$SK_6 = f(Z_6) = f(d_1^{-1} M_{1,6}) = f(d_2^{-1} M_{2,6}) = f(d_3^{-1} M_{3,6}) = f(d_4^{-1} M_{4,6}) = f(d_6^{-1} M_{6,6})$

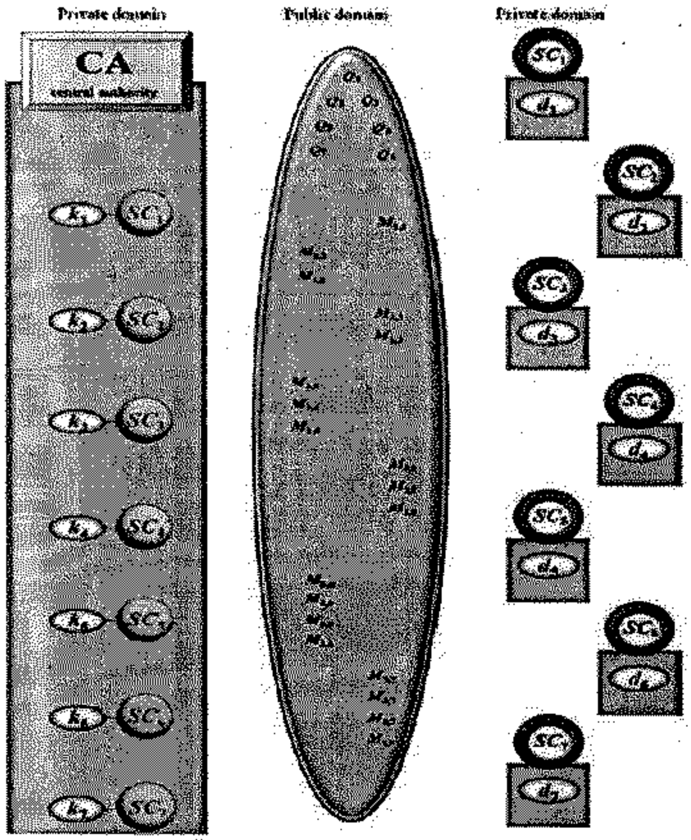
$SK_7 = f(Z_7) = f(d_1^{-1} M_{1,7}) = f(d_4^{-1} M_{4,7}) = f(d_7^{-1} M_{7,7})$

Table 2. Required Storage

The CA's private domain	*SCi's private domain*	***Public domain***
ki (ε Z_n) for $i = 1, 2 \ldots N$	*di* (ε Z_n)	**Q_i, $M_{i,j}$ for $i = 1,2,\ldots N$, $j = 1,2,\ldots \omega$ denotes the of SCi's successors(including SCi)**

In the table 2 the Nikhooghadam et al.'s [10] scheme, the central authority saves the value of the private parameters from k1 to k7. Every class in the hierarchy from SC1 to SC7 only saves the secret parameter d1 to d7. The parameters Mi,j and Qi to continue in the area called public domain. So everyone can access its content and no security is needed. In the fig (7) shows that all the points are secret by the SC and CA in public domain recorded all the parameters. Just needed the simple storage space not having any security agreement.

Figure 8

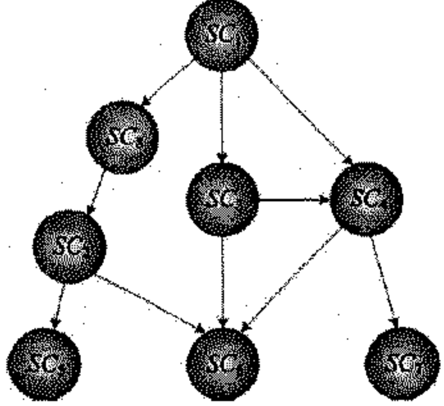


2.3.3 Adding new security class in the Nikhooghadam et al.'s scheme.

Into the hierarchy a new class SCx is added, such that $SCj <= SCx <= SCi$, the procedure is:

- *Step1:* Selects the $dx \in Z^*n$ random integer and its secret points for the security class *SCx, then* $Qx = dxP$ is calculated and its public parameter then announce them publically.
- *Step2:* CA chooses the $kx \in Z^*n$ random integer, so the *SCx* secret key is $SKx = H(Zx) = H(kxP)$.
- *Step3:* for every *SCi* satisfies the $SCx < SCi$.
  - Central authority decides the points $Mi,x = kxQi$ then declare publically.
- *Step 4:* for all security classes *SCj* satisfy $SCj <= SCx$.
  - CA decides all the points $Mx,j = kjQx$ and declare publically.

Figure 9, after adding new class SC8



To explain the above algorithm steps let's take an example, as shown in the fig (8) new security class is inserted a SC_8 into the hierarchy and $SC_2 <= SC_8 <= SC_1$. For new security class SC_8, CA chooses an integer which is random $k_8 \in Z^*n$, so SC_8 private key is $SK_8 = H(Z_8)$, central authority decides $M_{1,8} = k_8(Q_1)$, $M_{2,8} = k_2(Q_8)$, $M_{5,8} = k_5(Q_8)$, $M_{6,8} = k_6(Q_8)$ and $M_{8,8} = k_8(Q_8)$ because SC_8 give a successor to SC_1 and predecessor to SC_2, so SK_8 is gets through SC_1 and private keys $\{SK_2, SK_5, SK_6, SK_8\}$ will derive through SC_8.

2.3.4 Removing the security classes:

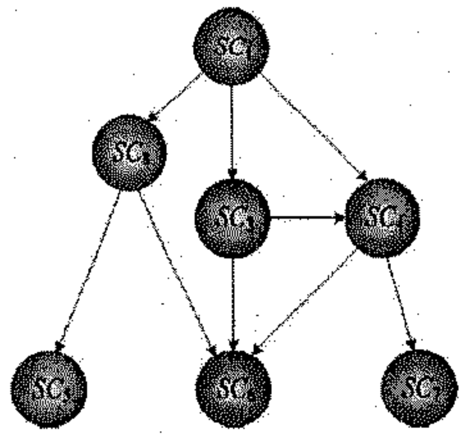
When remove the SCx from the user hierarchy the relationship SCi >= SCx >= SCj divides. So the SCx have no favor to access the authorized information, to achieve the control over forward security of every security classes SCj to satisfies the SCj < SCx. Central authority refresh all the private keys of SKj as SK*j., as shown below:

Step 1: SCj satisfy all the security classes, SCj < SCk.

The central authority again selects integer which is random $k^*j \in Z^*n$ and new private key of SCj is $SK^*j = H(Zj)$. SCi satisfies the $SCj <= SCi$ ($\neq SCx$) for every security class. CA decide the $Mi,j = k^*j(Qi)$ then show them publically.

Explain the earlier algorithm take an example; let's consider SC_2 is deleted from the user hierarchy as shown in the fig (9).

Figure10, deleting the SC2 from the hierarchy



CA deleted SC2 all parameters and creates the new user hierarchy as shown above in the fig 2(h). Control the forward security of SC_5 and SC_6, central authority again selects the two integers that is random k^*_5 and $k^*_6 \in Z^*n$, the SC_5 and SC_6 new private keys are $SK^*_5 = H(k^*_5P)$ and $SK^*_6 = H(k^*_6P)$, after that central authority decides $M_{8,5} = k^*_5(Q_8)$, $M_{5,5} = k^*_5(Q_5)$, $M_{1,5} = k^*_5(Q_1)$, $M_{1,6} = k^*_6(Q_1)$, $M_{3,6} = k^*_6(Q_3)$, $M_{4,6} = k^*_6(Q_4)$, $M_{6,6} = k^*_6(Q_6)$ and $M_{8,6} = k^*_6(Q_8)$ and also declare publically.

2.3.5 Drawbacks of Nikhooghadam et al.'s scheme:

- Few drawbacks in Nikhooghadam et al.'s [10] scheme which can be discussed one by one.CA have large private storage space, Central authority CA also saves the ki secret parameters for every security classes to compute the Zi and SKi whenever its possible, for example in the fig *2(b)*, CA saves in its private domain the key k_1 to k_7. And it is very difficult to maintain all parameters for the security classes.
- In the public domain require a huge storage space, to maintain public parameters in the public domain are points in the EC then needed huge storage space. Affect the performance and important factor of implementation of physical distribution of these parameters. Using the point compression method [13] in the public domain decrease the storage space. It enforces huge calculations cost for the scheme and high quality includes not in the private domains. So the Nikhooghadam et al.'s [10] scheme claim employment using method of compression is not needed in public domain.

- In the point multiplication process, encryption and decryption tare performed on the SKi secret keys based on point multiplication, central authority and every SC has to do many point multiplication process which encode and decode those private keys. E.g. central authority has to perform many point multiplications operation to calculate the Mi,j = kj(Qi) all the public parameters as shown in the fig (8). The secret class SC_2 wants to derive the secret keys which is authorized of his successors (SK6, SK5, SK2) between the similar public points (M2,6, M2,5, M2,2) in public domain using the point multiplications operation. In spite the fact that ECC is praiseworthy in terms of efficiency and security in comparing to other public key cryptosystem.

2.4 Basu et al.'s scheme:

In the Basu et al.'s [15] presented a cloud storage scheme using elliptic curve cryptosystem based on key management in user hierarchy. It contains the setup phase and key generation phase different private key of SC.

2.4.1 Setup Phase:

The setup phase contains the five steps and each step is discussed below:

Step 1: A trusted dealer (TD) chooses a secure EC, $GF(p)$ p belongs to prime and G is a base point of order q and $163 <= q$ bits.

Step 2: the trusted dealer selects his d_{TD} secret key and $dTD \in [1, q - 1]$and P_{TD} public key where $P_{TD} = d_{TD}.G$. TD also chooses di secret key where $dTD \in [1, q - 1]$and also public key as well. $Pi = di.G$for the SCi where $n >= I >= 1$. The SCi secret keys are divided through some secure channel between the members of SCi.

Step 3: TD calculates the $Zi = ki.G$ and ki is a random integer selected between $[1, 1- q]$ for every SCi and the trusted dealer also calculates the $Ski = H(Zi)$ key where H hash function and changes the x-coordinates of Zi on the EC through the private key Ski.

Step 4: SC satisfies SCi >= SCj from the interval 1 <= j <= n, TD calculates Yi,j = kj.Pi and give them Yi,j to other SC's using some safe way.

Step 5: Now trusted dealer announces (p, q, G, Pi, P_{TD}) and H hash function and also contain the d_{TD} secret key as well as SKi, the ki for every SC's in the secret place but delete di private keys of the SC.

2.4.2 Key Generation Phase:

Step 1: Every SC calculates the reverse of his/her key di which is private i.e. $d^{1}i$ and securely saves it.

Step 2: Then every SCi calculates $Zi = d^{1}i.Yi,i$ for his and $Zj = d^{1}i.Yi,j$ for Security classes SCj (SCi >= SCj, n >= j >= 1) as Zj = kj. Yi,j. So that will satisfies

$Y_{i,j} = kj.\ Pi = kj.\ (di.\ G)$

$d^{1}i.\ Yi,j = d^{1}i.(kj.\ (di.G)) = kj.\ G = Zj$

Where $Zj = d^{1}i.\ Yi,j$

Step 3: SCi calculates the private key $SKi = H(Zi)$ and H is hash function and changes the x-coordinates the point Zi on the EC to the secret key SKi. Now take the let's show that *SC3* have the right to get all data of the security classes *SC2, SC5, SC6, SC7, and SC8* as shown in the *fig 2(i)*, which is lower in the hierarchy and generates the keys of security classes of {SK2, SK5, SK6, SK7, SK8}. Yi,j has sent to SC3 by TD.

SC3: $Y_{3,2} = k_2.P_3$

$Y_{3,5} = k_5.P_3$

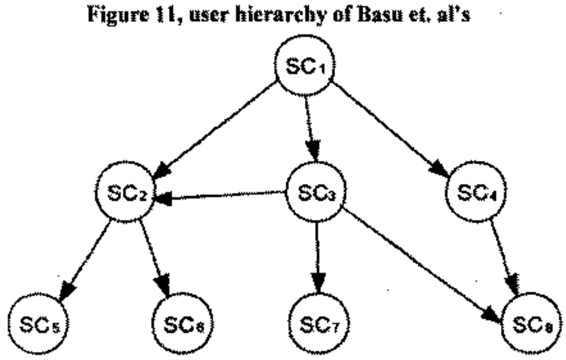
$Y_{3,6} = k_6.P_3$

$Y_{3,7} = k_7.P_3$

$Y_{3,8} = k_8.P_3$

$Y_{3,3} = k_3.P_3$

Figure 11, user hierarchy of Basu et. al's



Now SC3 can also get the security keys of the SC's also its own secret key

$SK_2 = H(Z_2) = H(d^1{}_3.\ Y_{3,2})$

$SK_5 = H(Z_5) = H(d^1{}_5.\ Y_{3,5})$

$SK_6 = H(Z_6) = H(d^1{}_6.\ Y_{3,6})$

$SK_7 = H(Z_7) = H(d^1{}_7.\ Y_{3,7})$

$SK_8 = H(Z_8) = H(d^1{}_8.\ Y_{3,8})$

$SK_3 = H(Z_3) = H(d^1{}_3.\ Y_{3,3})$

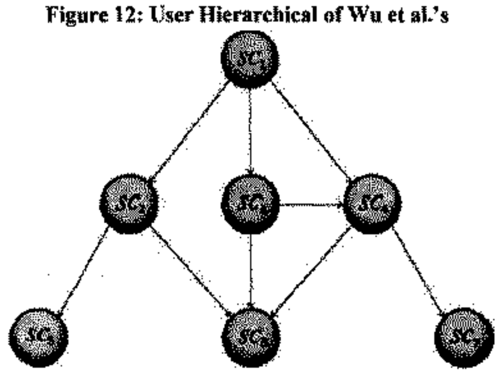
2.4.3 Changing Private Key of a Security Class:

Each security class SCj require to change his private key SKi and the TD chooses the $k^*{}_j$ from $[1, q - 1]$ then calculates the $SK^*j = H(k^*{}_j.\ G)$. All the security classes that satisfy $SCi >= SCj$ and TD calculates $Y^*{}_{i,j} = k^*{}_j.\ Pi$.

2.5 Wu et al.'s scheme

In this [15] scheme presents heterogeneous cryptosystem, asymmetric and symmetric scheme. The symmetric encryption is denoted by Ω and his encoding/decoding algorithm $Ek(\cdot)/Dk(\cdot)$ here k means symmetric key. The asymmetric remain as ECC because of its benefits of size constrains and processing. As like other proposed scheme Wu el al.'s scheme also uses the CA central authority to construct the access control of hierarchy in order to the relationship between SC's. Wu et al.'s [15] scheme it contains the key generation and key derivation phase discussed as below.

Figure 12: User Hierarchical of Wu et al.'s



2.5.1 Key Generation Phase

Step 1: The central authority decides the points *(E, G, H, Ω)* then declares it publically.

Step 2: every security class SCi chooses an integer which is random $di \in Z^*n$ and his private point, to calculate $Qi = diP$ as his public parameter then declare them publically.

Step 3: central authority chooses an integer which will be random $d_{CA} \in Z^*n$ and its secret key, calculate $Q_{CA} = d_{CA}P$ which is public and declare them publically. The central authority also chooses k_{CA} random bit string which his private key, to get the encrypted key $EKi = H(kCA \| SCi)$

For every security class SCi. So at the end central authority chooses a *SKi* random integer as secret key for every SC. The d_{CA} and k_{CA} are in a central authority in its secret domain. For each security class *EKi* and *SKi* secret keys would be removed due to the security reasons at the end of this phase.

Step 4: Each security class SCj (N >= j >= 1), central authority CA calculates $Zj = d_{CA}Qj$ and $CKj = H(Qj \| Q_{CA} \| Zj)$ and used to encrypt *EKi* as $Rj = E_{CKj}(EKj)$ declare *Rj* publically. For every SCi satisfy $SCi >= SCj$. The CA decides $M_{i,j} = E_{EKi}(SKi)$ declare them publically. In Wu et al.'s [15] scheme Qi, Q_{CA}, $M_{i,j}$ and *Ri* the public parameters are saves into public domain and can be access using every users. The private point's d_{CA}, k_{CA} and di are secret in the private domain. CA cannot store any private parameters of *SKi* for the SC's since it recovers from public parameter $M_{i,j}$ employing the private key k_{CA} whenever needed. First of all the CA uses k_{CA} to get the EKi then takes the final value to decode $M_{i,j}$ by $SKi = D_{EKi}(M_{i,j})$. Every encrypted key EKi able to build using the secret key k_{CA} and the identity of same SC.

2.5.2 Key Derivation Phase

The relation SCi >= SCj the SCi decides private keys of all his successors SCj and private key *SKi* as shown below.

Step 1: $Zi = diQ_{CA}$ and calculates $CKi = H(Qi \| Q_{CA} \| Zi)$.

Step 2: decides the $EKi = D_{CKi}(Ri)$ and compute $SKj = D_{EKi}(M_{i,j})$

Step 3: every security class SCi have private parameter di to derive the private key of the successors.

Example to show the hierarchy in the fig (12) as shown above, that set of needed points for deriving the private keys of every security classes of his predecessors.

- SC1: CA calculates the $R_1 = E_{CK1}(EK_1)$ *and* $M_{1,1} = E_{EK1}(SK_1)$
- SC2: CA calculates the $R_2 = E_{CK2}(EK_2)$ *and* $M_{1,2} = E_{EK1}(SK_2)$, $M_{2,2} = E_{EK2}(SK_2)$
- SC3: CA calculates the $R_3 = E_{CK3}(EK_3)$ *and* $M_{1,3} = E_{EK1}(SK_3)$, $M_{3,3} = E_{EK3}(SK_3)$
- SC4: CA calculates the $R_4 = E_{CK4}(EK_4)$ and $M_{1,4} = E_{EK1}(SK_4)$, $M_{3,4} = E_{EK3}(SK_4)$, $M_{4,4} = E_{EK4}(SK_4)$
- SC5: CA calculates the $R_5 = E_{CK5}(EK_5)$ *and* $M_{1,5} = E_{EK1}(SK_5)$, $M_{2,5} = E_{EK2}(SK_5)$, $M_{5,5} = E_{EK5}(SK_5)$
- SC6: CA calculates the $R_6 = E_{CK6}(EK_6)$ *and* $M_{1,6} = E_{EK1}(SK_6)$, $M_{2,6} = E_{EK2}(SK_6)$, $M_{3,6} = E_{EK3}(SK_6)$, $M_{4,6} = E_{EK4}(SK_6)$, $M_{6,6} = E_{EK6}(SK_6)$
- SC7: CA calculates the $R_7 = E_{CK7}(EK_7)$ *and* $M_{1,7} = E_{EK1}(SK_7)$, $M_{3,7} = E_{EK3}(SK_7)$, $M_{4,7} = E_{EK4}(SK_7)$, $M_{7,7} = E_{EK7}(SK_7)$

Now every SC derives his private key and secret keys of his successors as shown below:

- SC_1: $Z_1 = d_1 Q_{CA}$, $CK_1 = H(Q_1 \| Q_{CA} \| Z_1)$, $D_{CK1}(R_1)$, $SK_1 = D_{EK1}(M_{1,1})$, $SK_2 = D_{EK1}(M_{1,2})$, $SK_3 = D_{EK1}(M_{1,3})$, $SK_4 = D_{EK1}(M_{1,4})$, $SK_5 = D_{EK1}(M_{1,5})$, $SK_6 = D_{EK1}(M_{1,6})$ $SK_7 = D_{EK1}(M_{1,7})$.
- SC_3: $Z_3 = d_3Q_{CA}$, $CK_3 = H(Q_3 \| Q_{CA} \| Z_3)$, $EK_3 = D_{CK3}(R_3)$, $SK_3 = D_{EK3}(M_{3,3})$, $SK_4 = D_{EK3}(M_{3,4})$, $SK_6 = D_{EK3}(M_{3,6})$, $SK_7 = D_{EK3}(M_{3,7})$.
- SC_4: $Z_4 = d_4Q_{CA}$, $CK_4 = H(Q_4 \| Q_{CA} \| Z_4)$, $EK_4 = D_{CK4}(R_4)$, $SK_4 = D_{EK4}(M_{4,4})$, $SK_6 = D_{EK4}(M_{4,6})$, $SK_7 = D_{EK4}(M_{4,7})$.
- SC_5: $Z_5 = d_5Q_{CA}$, $CK_5 = H(Q_5 \| Q_{CA} \| Z_5)$, $EK_5 = D_{CK5}(R_5)$, $SK_5 = D_{EK5}(M_{5,5})$.
- SC_6: $Z_6 = d_6Q_{CA}$, $CK_6 = H(Q_6 \| Q_{CA} \| Z_6)$, $EK_6 = D_{CK6}(R_6)$, $SK_6 = D_{EK6}(M_{6,6})$.
- SC_7: $Z_7 = d7Q_{CA}$, $CK_7 = H(Q_7 \| Q_{CA} \| Z_7)$, $EK_7 = D_{CK6}(R_7)$, $SK_7 = D_{EK7}(M_{7,7})$.

CA only save the secret parameters of k_{CA} and d_{CA}. Every security class SC1 to SC7 only saves his private parameter from d1 to d7. Q_{CA}, Qi, Mi,j and Ri should be in public domain.

2.5.3 When new security classes is added in the hierarchy

The SCx is added in to user hierarchy so SCi >= SCx >= SCj. And the SCx in the user hierarchy can be control by the following method:

Step 1: SCx chooses a random number $dx \in Z*n$ and his private points, and the $Qx = dxP$ is calculated and then declare them publically.

Step2: central authority CA chooses a random integer $kx \in Z*n$. so SCx private key is SKx = H(Zx) = H(kxP).

Step 3: every security classes SCi satisfy SCi >= SCx

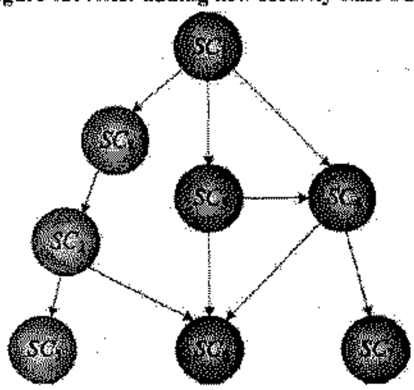
- CA decides the Mi,x = kxQi and declared them publically.

Step 4: every security classes SCj satisfy SCx >= SCj

- CA decides the points Mx,j = kjQx and declared them publically.

To explain the algorithm given above let's take an example as shown in the fig 2(k).

Figure 13: After adding new security class SC8



After adding a new class SC_8 in to user hierarchy as in the *fig 2(k)*, the CA chooses random integer $k8 \in Z*n$. so the private key of SC_8 is $SK_8 = H(Z_8)$, and central authority decides $M_{1,8} = k_8(Q_1), M_{8,2} = k_2(Q_8), M_{8,5} = k_5(Q_8), M_{8,6} = k_6(Q_8),$ *and* $M_{8,8} = k_8(Q_8)$ because SC_8 given a successor to SC_1 and a predecessor to SC_2. So SK_8 is derived by SC_1 and the secrets keys can be derived through SC_8 (SK_2, SK_5, SK_6, SK_8).

2.5.4 Removing the Security Class from the hierarchy

When *SCx* is deleted from the hierarchy then the relation will be *SCi >= SCx >= SCj*, so the *SCx* having no favor to get whole data that primarily was right to access. The forward security is controled by every security classes *SCj* which satisfy *SCx >= SCj*. And now the CA again new the entire private keys *SKi* as *SK*j* as shown below.

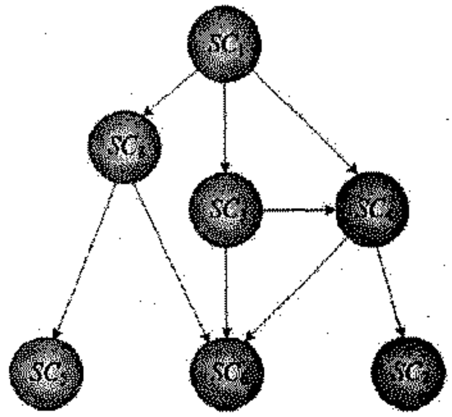
Step 1: every *SCj* will satisfy the *SCx > SCj*.

CA again chooses a random number $k^*j \in Z^*n$. So new private key of *SCj* is $SK^*j. = H(Zj)$.

- Every security classes *SCi* which satisfy *SCx >= SCj* (≠*SCx*).

- CA decides the points $M_{i,j} = E_{EK_i}(SK^*_j)$ and declare it publically.

Figure 14: after deleting the SC2 from user hierarchy



To explain the algorithm above lets takes an example, suppose from the user hierarchy SC_2 is removed as shown in the *fig2 (l)*. CA has removed all the parameters which are connected to SC_2 and change the ways of SC_2. So to control the forward security of SC_5 and SC_6, central authority again selects two random bit string SK^*_5 and SK^*_6 as the new private key of SC_5 and SC_6 and after this CA decides $M_{8,5} = E_{EK8} (SK^*_5)$, $M_{5,5} = E_{EK5}(SK^*_5)$, $M_{1,5} = E_{EK1}(SK^*_1)$, $M_{1,6} = E_{EK1} (SK^*_6)$, $M_{3,6} = E_{EK3} (SK^*_6)$, $M_{4,6} = E_{EK6} (SK^*_6)$, $M_{6,6} = E_{EK6} (SK^*_6)$, and $M_{8,6} = E_{EK8} (SK^*_6)$ and then declare publically.

2.5.5 Drawbacks of Wu et al.'s scheme:

In this [15] paper has some disadvantages which are discussed as follows,

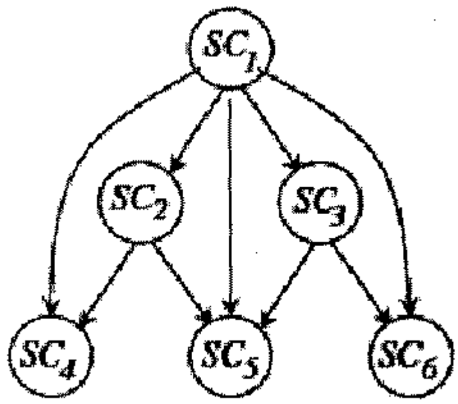
- The scheme requires huge space for storage in public domain. QCA and Qi where 1 <= i <= N are the points on elliptic curve Ep(a, b) and stored in the public domain. For example the hierarchy size is very large, so physically distribution of the parameters affects of doing something and can have great value factor in implementation. And the public value Ri and Mij for (1 <= I <= N) and 1 <= j <= vi, where vi is the successors of SCi and require to store in the public domain. So there could be millions of SC in the user hierarchy and every security key is gven to thousands of security class [10].
- Elliptic curve use of costing multiplication and the encryption/decryption of the key EKi and every security class SCi needs to compute the point Zi = diQ_{CA} in to compute the CKi = H(Q_{CA}QiZi). so the point multiplications of the elliptic curve are expensive and

method of derivation and generation of key is inefficient. For deleting and adding the security classes and to make the different relation Wu et al.'s [15] require to update a huge number of secret keys and size of the user hierarchy is large and at the end the dynamic access problem will be become slow.

2.6 Review of Chen et al.'s scheme

Chen et al.'s [17] uses the novel key management scheme based on one way hash function. This method contains two phases, 1st key generation and 2nd key derivation phase.

Figure 15: user hierarchy of Chen et al.'s scheme



2.6.1 Key Generation Phase:

Let's consider there are n number of classes i.e. SC1, SC2, SC3,.......SCn in hierarchy, central authority do the mentioned steps to calculate the private key ski for the security class SCi (1, 2, 3, ...n).

Step 1: chooses the symmetric cryptosystem in which $E_k(\cdot)$ and $D_k(\cdot)$ for the encryption and decryption algorithm with key *k*.

Step 2: chooses hash function $H(\cdot)$ and declare it publically.

Step 3: each SCi (i = 1, 2, 3, 4.....,n) in user hierarchy, central authority randomly chooses two large positive integers Pi and ski and so central authority sends ski to SCi in safe channel then declare publically Pi.

Step 4: the preorder traversal take the security class SCi from user hierarchy.

Step 5: every SC SCj from the user hierarchy having relation $SCi >= SCj$, then central authority calculates the similar public parameter $Rij = E_{H(Pj \oplus ski)}(skj)$. ($\oplus$ implies the bit wise exclusive OR operation).

Step 6: step 4 should be repeated when every SC in the user hierarchy.

2.6.2 Key Derivation Phase.

Suppose SCI is the predecessor of SCj so the $SCi >= SCj$. To get back the data which is secret to SCj, SCi first gets the private key skj. And the method is for deriving the skj as shown below.

Step 1: first take the private key ski of SCi and public parameter Pj of SCj to calculate the hash value $H(Pj \oplus ski)$.

Step 2: decide the public parameter Rij of SCi and SCj. Then derive the $skj = D_{H(Pj \oplus ski)}(Rij) = D_{H(Pj \oplus ski)}(E_{H(Pj \oplus ski)}(skj))$.

2.6.3 Addition a new security class in the hierarchy

Suppose SCk is newly added into the user hierarchy and the procedure is shown below in the algorithm.

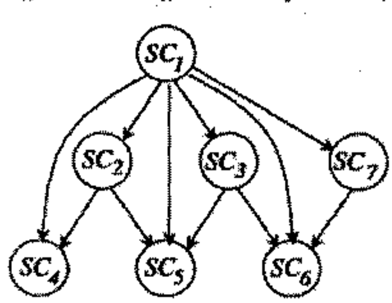
Step 1: Central authority CA randomly chooses two integers Pk and Sk_k and then send Sk_k to SCk through secure channel and Pk declare it publically.

Step 2: each SCi with the relation SCi >= SCj and central authority calculates the public parameter Rik, where $Rik = E_{H(Pk \oplus ski)}(sk_k)$.

Step 3: every SCj with the relation SCk >= SCj and CA calculates the public parameters Rkj where $Rkj = E_{H(Pj \oplus skk)}(sk_j)$.

Step 4: each SCi and SCj with a relation $SCi >= SCk$ and $SCk >= SCj$. Central authrity calculates the public points Rij where $Rij = E_{H(Pj \oplus ski)}(sk_j)$.

Figure 16: adding new security class SC_7



Suppose in fig (16) is SC_7 included in the hierarchy, and to work as the successor of SC_1 and predecessor SC_6. Then CA randomly chooses pair of positive integers P_7 and sk_7. Then central authority calculates the public parameters $R_{1,7}$, $R_{7,6}$ for the recently added relation. So central authority updates three public parameters $R_{1,7}$, $R_{7,6}$ and P_7 in the public domain for the private key derivation.

2.6.4 Deleting Security Class from the user Hierarchy

For example SCk is the security class and removed from the hierarchy and update public parameter, the CA gain the cancellation of whole public parameters belonging to SCk, and no need to replace the related users keys. So the algorithm as shown below,

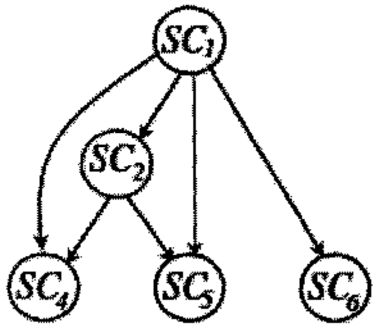
Step 1: central authority CA Pk and Sk_k

Step 2: central authority removes Rik for every security class SCi and the relation is SCi >= SCk.

Step 3: central authority Rij for every security class SCk and the relation is SCk >= SCj

Step 4: Central authority Rij for every SCi >= SCj and SCk >= SCj and the relation SCi >= SCj.

Figure 17: SC3 is deleted from the user hierarchy.



For example the SC_3 is removed from the hierarchy as shown in the fig 2(o). And CA has finish the registration of $R_{1,3}$, $R_{3,5}$, $R_{3,6}$ and sk_3 recorded.

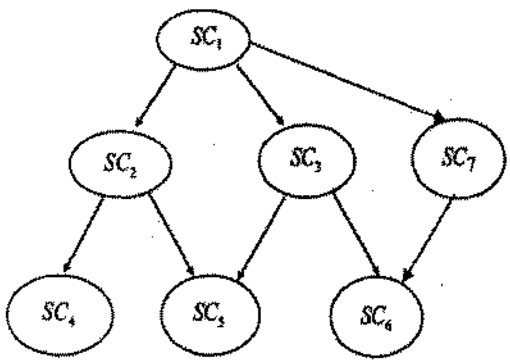
2.7 Review of Lin- Hsu scheme:

In Lin- Hsu scheme [18] has three phase's initialization phase, key generation and key derivation phase.

2.7.1 Key Initialization Phase:

In key initialization phase central authority selects a prime p which will be large also q and $a, b \in Z * p$ is two points to satisfy the $4a^3 + 27b^2 \bmod p \neq 0$. Suppose $Eq(a, b)$ be an EC $GF(p)$ contains a set of points $(x,y) \in Z * p$ and a parameter O at infinity, where $y^2 = x^3 + ax + b \pmod p$. Suppose $G1$ be additive cyclic group with prime order q and G be a generator of $G1$.Central authority chooses symmetric cryptosystem in which encryption/decryption $Ek()$ and $Dk()$ are two algorithm with key k. CA chooses two secure one way hash function $H1: \{0, 1\} \times G1 \rightarrow Z^*q$ and $H2: G1 \rightarrow Z^*q$. Central authority CA decides his secret key kca and makes Yca and makes its public $Ep(a, b, G, Yca$.

Figure 18: Partially ordered user hierarchy



2.7.2 Key Generation Phase:

In the 2nd phase the suppose the security class $SC = \{SC1, SC2\ SC3, \ldots\ldots SCn\}$ is in the hierarchy with disjoint sets of SC's that are partially ordered of a binary relationship <=. Every SC SCi selects his private key ki, $1 \in Z * q$and calculates the public key $Yi = ki,1G$ and encryption key $ki,2$ for $i = 1, 2, 3, \ldots\ldots n$.

Step 1: Every security class $SCi \in SC$ do the following steps to create its secret information Vi:

- Selects randomly a number $ri \in Z^*q$ and calculate the public information $Ri\ riG$.
- Calculates the encrypted key $ki,2 = H1(ki,1, Ri)$.
- Calculate $Vi = k1,2\ (H2(ri\ Yca)) \bmod q$.
- And then send the secret information Vi to central authority using some secure channel.

Step 2: After receives the Vi form the SCi for $i= 1, 2, 3 \ldots n$, central authority CA calculates $ki,2 = Vi(kcaRi)$.

Step 3: central authority CA calculates the integer $DK_{i\rightarrow j} = H\,(ki,2,\ Yj)$ for all *SCi*'s where $j = 1, 2, 3, \ldots\ldots\ldots n$. so the *SCi* >= *SCj* and the symbol $DK_{i\rightarrow j}$ denoted for the key derivation of *SCi* to gets the encryption key of his successor *SCj*.

Step 4: Central authority employs the polynomial interpolation to decide a public function $fj(x)$ for every security class *SCj* where $j = n, n-1 \ldots\ldots 1$. So the polynomial $fj(x)$ put together through the points $(DK_{i\rightarrow j},\ EH1(ki,2,\ Rj)(kj,2))$ for every security class *SCi* for *SCi* >= *SCj*.

2.7.3 Key Derivation Phase:

In 3rd phase the security class SCi want to access the encoded information of SCj where SCi >= SCj, so SCi will do the require steps to get SCi's encryption key kj,2.

Step 1: calculates the derivation key points $(DK_{i\rightarrow j} = H1(ki,2,\ Yj)$.

Step 2: calculates the $EH1\ (ki,2,\ Rj)(kj,2)) = fj(DK_{i\rightarrow j})$.

Step 3: Get the encryption key kj,2 of SCj by decrypting $EH1\ (ki,2,\ Rj)(kj,2))$ as $DH1\ (ki,2,\ Rj)(\ EH1\ (ki,2,\ Rj)(kj,2))$.

2.7.4 Adding a new Security Class:

For example *SCl* is included new in to the hierarchy so *SCi* > *SCl* > *SCj*. The *SCl* calculates his private key $kl,2$, and generates the private data *Vl* and send to central authority and also updates the function $fj(x)$ that is public of security class *SCj (SCl < SCi)*.

Step 1: SCl do the following steps to generate his own secret information *Vl*,

- Selects randomly a number $rl \in Z^*p$ and calculates the public information $Rl = rlG$.
- Calculates the $Vl = kl,2\ (H2(rlYca))\ mod\ q$.
- And then sends the secret information *Vl* to central authority through some secure channel.

Step 2: After receiving the *Vl* from *SCl*, central authority calculates the $kl,2 = Vl\ (H2(kcaRl))^{-1}$ mod q to take out *kl,2* by using its private key *kca*.

Step 3: every security class *SCl*'s where *SCl* > *SCj*, and CA calculates points $(DK_{l\rightarrow j} = H1(kl,2,\ Yj)$ and to build new polynomial $fj(x)$'s considering $(DK_{l\rightarrow j} = EH1(kl,2,\ Rj(kj,2))$ points.

Step 4: every security class SCi's where $SCl < SCi$ and security class calculates $DK_{i \rightarrow l}, = H1(kl,2, Yl)$, $(DK_{l \rightarrow i}, = H1(ki,2, Yl)$ and build new polynomial fl(x)'s including the points $(DK_{i \rightarrow l}, = EH1(ki,2, Rl)(kl,2))$'s.

2.7.5 Deleting the Security Class:

Suppose SCl class is removed from the hierarchy so $SCi > SCl > SCj$ and central authority CA construct the SCl and also relation between predecessor SCi and successor SCi of SCl. Central authority do the following steps,

Step 1: every security class SCj's from the hierarchy and delete the private key and public parameter of SCl.

2.8 Problem statement:

All the existing schemes for key management in hierarchal access control are not secure against exterior roots finding attack which can be launched after addition or deletion of some security class and updation of relationship between the new classes, some suggestion have been proposed by (ASOK et al) and Hsu et al, but these require more memory and computation power.

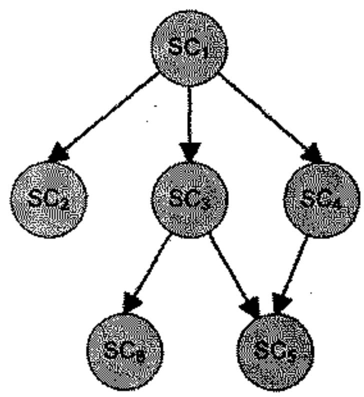
Chapter 3

Proposed Scheme

3. Proposed Scheme:

The proposed scheme contains the initialization phase, key generation phase and key derivation scheme. In key derivation phase, a central authority decides all system parameters. Key generation phase, every SC selects its private key and calculates matching public key. After that every SC use its private key and public parameter to calculate his encryption key. After that each security class sends its encrypted key to CA. central authority derives the encrypted key of the SC and constructs the public polynomial of every SC. In the 3rd phase, the predecessor can use his encryption key and the public information belonging to the successor and gets the decrypted key for accessing the authorized files. Whole description of these phases is given below.

Figure 19: partially ordered user hierarchy



3.1 First key initialization phase:

Central Authority (CA) chooses prime p which will be large and EC, $E_p\,(a, b)$: $y^2 = x^3 + ax + b$ *mod p* with a point O at infinity, where $a, b \in Z^*_p$ are two random integers satisfying that $4a^3 + 27b^2$ *mod p* not equal to 0.Suppose $G \in E_p\,(a, b)$ be a base point of order q, where q is a high prime. Central authority also chooses a transformation function $\tilde{A}: (x, y) \rightarrow z$ to transform a point on $E_p\,(a, b)$ into a real number $z \in Z^*_p$, at last the central authority announces the $(p, q, \tilde{A}, E_p(a, b), G)$.

3.2 Key generation phase:

Central authority selects randomly the private parameter Y_{ca} and makes the $K_{ca} = Y_{ca}G$ public. Each SCi have Y_i and r_i secret where $r_i \in Z^*_{q.}$ $R_i = r_iG$ public

$K_i = H(R_iY_{ca})$.

SC_i send **ri** to CA via some secure channel and CA received *ri*, to calculate Ki.

$$K_i = H(R_iY_{ca})$$
$$K_i = H(r_iG\ Y_{ca})$$
$$K_i = H(r_i\ Y_{ca}G)$$
$$K_i = H(r_i\ K_{ca})$$

3.3 Key Derivation phase:

The predecessor SC_i able to compute all his successor(s) (SC_j) keys of S_k for which the relationships $(SC_i, SC_j) \in R_{ij}$ between the SC_i and SC_j hold as follows:

$$K_i = P_i(x) = H(\Pi_{sci>scj}(x - (R_{ij})) + r_iK_{ca})$$
$$K_1 = P_1(x) = Nil$$
$$K_2 = P_2(x) = H(\Pi(x - (R_{21})) + r_2K_{ca})$$
$$K_3 = P_3(x) = H(\Pi(x - (R_{31})) + r_3K_{ca})$$
$$K_4 = P_4(x) = H(\Pi(x - (R_{41})) + r_4K_{ca})$$
$$K_5 = P_5(x) = H(\Pi(x - (R_{54}))(x - (R_{53}))(x - (R_{51})) + r_5K_{ca})$$
$$K_6 = P_6(x) = H(\Pi(x - (R_{63}))(x - (R_{61})) + r_6K_{ca})$$

For example SC_1 wants to derive his successor SC_6 secret key H $(r_6\ K_{ca})$ to encrypt the information.

$$P_6(x) = H(\Pi(x - R_{63}))(x - (R_{61})) + r_6K_{ca})$$

$$R_{61} = H(R_6\ Y_1)$$
$$R_{61} = H(r_6G\ Y_1)$$
$$R_{61} = H(r_6\ Y_1G)$$
$$R_{61} = H(r_6\ K_1)$$

$P_6(x) = H(\prod(x - (R_{6\,3}))(x-(R_{6\,1})) + r_6K_{ca})$

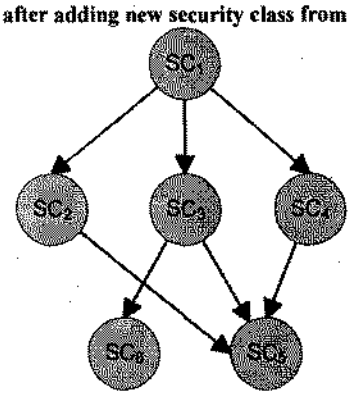
$P_6(\boldsymbol{R_{6\,1}}) = \boldsymbol{H}(\prod((\boldsymbol{R_{6\,1}}) - (R_{6\,3}))\ (\cancel{(R_6K_1)} - \cancel{(R_{6\,1})}) + r_6K_{ca})$

$P_6(\boldsymbol{R_{6\,1}}) = \boldsymbol{H}(r_6K_{ca}) = H(R_6Y_{ca})$

3.4 Adding a new Security Class:

Suppose a new SC is included in the user hierarchy as shown in the fig (19). As the new SC is updated from SC_2 to SC_5 , so the secret keys will be update in the hierarchy for SC_5 as shown in the eq (3.2).

Figure 20: after adding new security class from SC2 to SC5



$$P_5(x) = H(\prod (x - (R_{5\,4}))\,(x - (R_{5\,3}))\,(x - (R_{5\,1})) + r_5K_{ca}) \quad (3.1)$$

$$P_5(x) = H(\prod (x - (R_{5\,4}))\,(x - (R_{5\,3}))\,(x - (R_{5\,2}))\,(x - (R_{5\,1})) + r_5K_{ca}) \quad (3.2)$$

So the secret key of SC_5 $K_5 = H(r_5K_{ca})$ can be derived and SC_5 also derive his successor's secret keys. For example SC_4 wants to derive the private key of its successor SC_5 using his own private parameter K_4 and the public parameter of SC_5 which r_5 as shown in the eq(3.3).

$P_5(x) = H(\prod (x - (R_{5\,4}))\,(x - (R_{5\,3}))\,(x - (R_{5\,2}))\,(x - (R_{5\,1})) + r_5K_{ca})$

$R_{5\,4} = H(r_5\,K_4)$

$R_{5\,4} = H(r_5\,Y_4G)$

$R_{5\,4} = H(r_5G\,Y_4)$

$R_{5\,4} = H(R_5\,Y_4)$

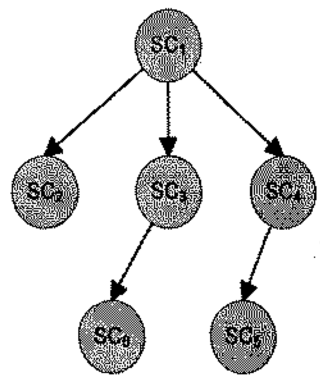
$$P_5(x) = H\left(\prod \left(\cancel{(R_{5\,4})} - \cancel{(R_{5\,4})}\right) ((R_{5\,4}) - (R_{5\,3})) ((R_{5\,4}) - (R_{5\,2})) ((R_{5\,4}) - (R_{5\,1})) + r_5K_{ca}\right) \quad (3.3)$$

$$P_5(x) = H\,(r_5K_{ca})$$

3.5 Removing a Security Class from the user hierarchy:

Consider a relation between SC3 and SC_5 is updated in the hierarchy as shown in the fig (20). Central authority updated all the parameter which is related to SC_5. To control the forward security of remaining SC(s) and also the secret key of SC5 is secured because the two different inputs to at least generates two different outputs and also having no correlation with each other. So SC5 is secured and is shown below eq (3.4).

Figure 21: after removing a security class from SC3 to SC5



$$P_5(x) = H\left(\prod (x - (R_{5\,4}))\,(x - (R_{5\,3}))\,(x - (R_{5\,1})) + r_5K_{ca}\right)$$

$$P_5(x) = H\left(\prod (x - (R_{5\,4}))\,(x - (R_{5\,1})) + r_5K_{ca}\right)$$

(3.4)

Chapter 4

Simulation and Performance Analysis

4. Simulation and Performance Analysis:

All the primitives were implemented in C language, where system specifications were processor Intel(R) core 2 duo, cpu 1.8 and 1.8 GHZ, Ram 2 GB, System type 32 bit Operating system. The time taken for each primitive is shown in the table 3 below. In this section compare the performance analysis of the proposed scheme with the Chung et al.'s [9], Chen-Huang's [4], Hsu et al.'s [5] and Jeng-Wing [6]. Table 4, shows the performance in terms of complexity for access control problems between the proposed scheme and previous schemes. For analysis the following notations are used as shown below.

n: number of SC(s) in user hierarchy

ki: degree of polynomial f(x)

d: number of direct parent nodes

$|x|$: the bit-length of an integer x

T_M: time for performing a modular multiplication

T_{INV}: time for performing the inverse element calculation

$T_{EC\,M}$: time for executing a scalar multiplication on the EC

$T_{EC\,ADD}$: time for executing an addition or subtraction on the EC

T_H: time for performing a one-way hash function H1 or H2

T_A: time for performing an algorithm to transfer a point on the EC as real number

Table 1. Computation time of primitive operations

	GF(2^512)	GF(2^160)
T_M	Negligible	Negligible
T_{INV}	220 μ sec	Negligible
$T_{EC\ M}$	2.578 sec	0.1246 sec
T_H	0.04 m sec	Negligible
T_{ADD}	0.0886 sec	0.00921 sec

The comparisons of computational and storage complexities of Jeng-Wing [6], Chung et al.'s [9], Hsu et al.'s [4], Lin-Hsu [18] and the proposed scheme are shown in the table 4 and 5. And from the table 4, it is clearly see that the proposed scheme is more efficient than Jeng-Wing scheme [6], Chung et al.'s [9], Hsu et al.'s [4] and Lin-Hsu [18] in saving the number of secret keys. In table 5 summaries the performance analysis of the Jeng-Wing [6], Chung et al.'s [9], Hsu et al.'s [4], Lin-Hsu [18] and proposed scheme. Consideration of the generation phase, the calculation efforts needed by central authority depends on the construction of an interpolating polynomial $fi(x)$ having degree k for SCi by applying Horner's rule; it needed k multiplications plus k additions for calculating a root of an interpolating polynomial with degree k. In Jeng-Wing scheme central authority spends $n(3T_{EC\ M} + 2T_{EC\ ADD})$ to calculate $\{rG, ki,2, ki,1) + rYca\}$'s then take out $(ki,2, ki,1)$'s and $\sum^{n}_{i=1} ki(T_{EC\ M} + T_M + T_A)$ to construct n polynomials. Every SC CA needed $(T_{EC\ M} + kiT_M)$ to get the successor's encrypted keys. So the total calculated amount of Jeng-Wing [6] is $(3n \text{ and } \sum^{n}_{i=1} ki(T_{EC\ M})) + 2nT_{EC\ ADD} + \sum^{n}_{i=1} ki(T_M + T_A)$. In Chung et al.'s [9] central authority spends $\sum ni=1\ ki(T_{EC\ M} + T_H)$ to calculate the $siGj = (x_{j,i} \| y_{j,i})$'s and $h(x_{j,i} \| y_{j,i})$'s

Table 2. Storage Complexity

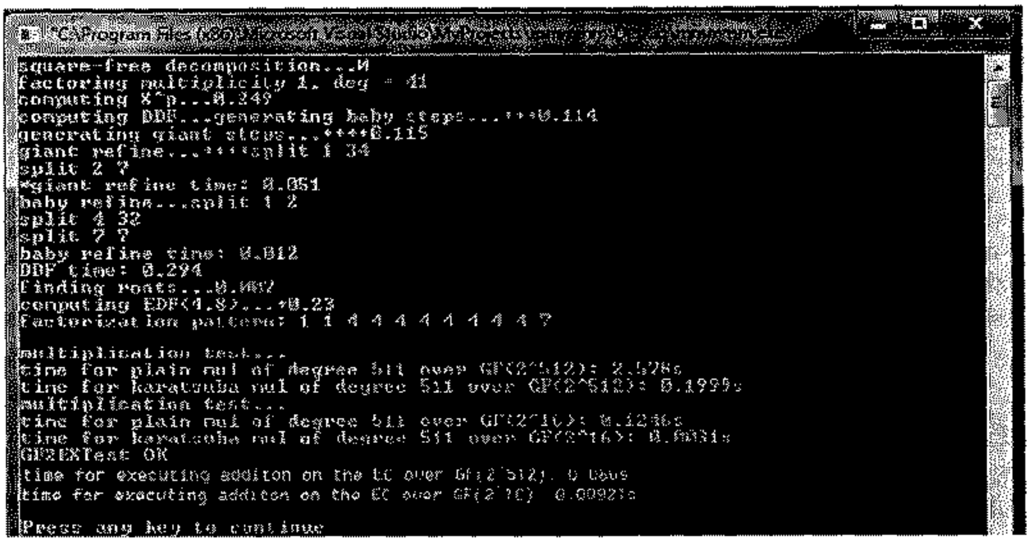
	Jeng-Wing [6]	Chung et al.'s[9]	Lin-Hsu[18]	Proposed scheme
Storage of public Parameters	$(3n+(\sum^{n}_{i=1} ki)+4)\|p\|$	$(2n+(\sum^{n}_{i=1} ki+3)\|p\|$	$(3n+(\sum^{n}_{i=1} ki)+4)\|p\|$	$(2n+(\sum^{n}_{i=1} ki+2)\|p$
Private keys for every SC	2|p|	|2|p	|p|	|p|

Table 3. Performance analysis

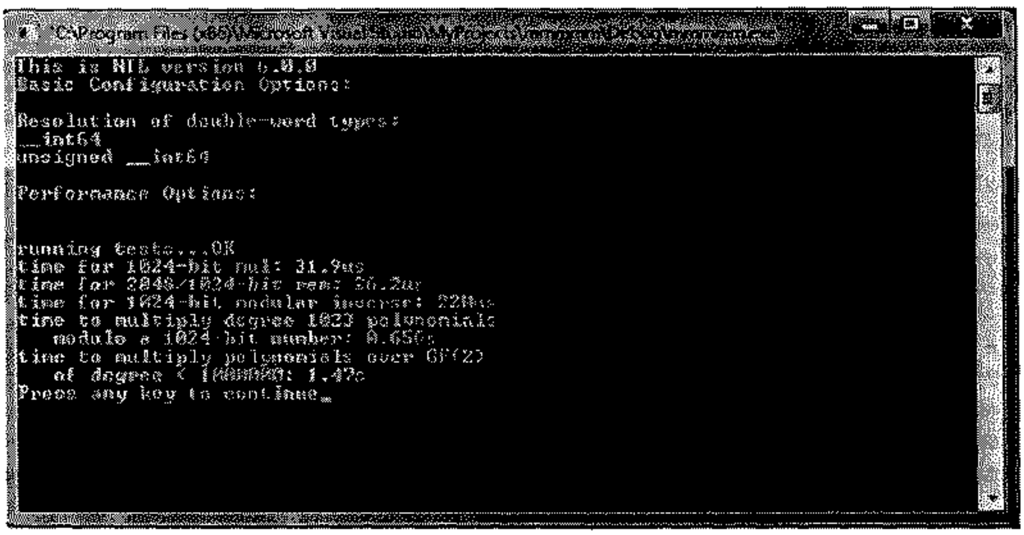
	Jeng-Wing [6]	Chung et al.'s [9]	Lin-Hsu [18]	Proposed scheme
Communication Cost	4n|p	N/A	n|p|	n|p|
Construction Of the public Polynomial f(x)'s for key derivation	$n(3T_{EC\,M}+2T_{EC\,ADD})+\sum^{n}_{i=1}ki(T_{EC\,M}+T_M+T_A$	$(2\sum^{n}_{i=1}ki(T_{EC\,M}))+\sum^{n}_{i=1}kiT_H$	$n(3T_{EC\,M}+2T_M+3T_H+T_{INV})+\sum^{n}_{i=1}ki(T_M+2T_H)$	$\sum^{n}_{i=1}ki(T_{ECM}+T_H)+n(T_H+T_{EC\,M})$
key derivation for a SC	$T_{EC\,M}+kiT_M$	$T_{EC\,M}+kiT_M+T_H$	kiT_M+2T_H	$kiT_M+T_H+T_{EC\,ADD}+T_{EC\,M}$

and $\sum^{n}_{i=1} ki\, T_{EC\,M}$ to construct n polynomials. Every SC needed $T_{CE\,M} + ki\, T_M + T_H$ to develop the successor's encryption keys. The total calculated amount of Chung et al.'s [9] is $(2n\sum^{n}_{i=1} ki +1)\, T_{EC\,M} + (\sum^{n}_{i=1} ki + 1)T_H + kiT_M$. In Lin-Hsu [18] central authority spends $(3nT_{CE\,M} + 2nT_M + 3nT_H + nT_{INV})$ to calculate secret data vi's and $ki,2$'s and $(\sum^{n}_{i=1} ki(T_M + 2T_H))$ to construct n polynomials. Every SC needed $(2TH + kiTM)$ to get the successor's encrypted keys. The total calculated amount of Lin-Hsu [18] is $\{3nT_{EC\,M} + (2n + ki + \sum^{n}_{i=1} ki)T_M + (3n + 2\sum^{n}_{i=1} ki + 2)T_H + nT_{INV}\}$. Also consider that the dynamic key management in adding a new SC, removing an existing SC, creating a new relation, to change an existing security class SC and updating a private key. In Jeng-Wing [6] every SC able to select his private keys$(k_{i,2}\, k_{i,1})$. The security class SCi employs his private key $k_{i,2}$ and SCj's public key Yj to calculate $\hat{A}(k_{i,1}\, Y_j)$, $SCi >= SCj$. So the SCi uses the fixed point $\hat{A}(k_{i,1}\, Y_j)$ to derive SCj's encryption key $k_{j,2}$. It is seen that the fixed point of every SC is the result of the predecessor's SCi public key. To make sure the forward and backward security of the successor's SCj, in spite adding or deleting an existing SC or to change an existing relationship the private key of the SC in Jeng-Wing [6], SCj has to update the private keys $k_{j,2}$ and CA require to update the related public parameters $f_j(x)$. In same situation the proposed scheme only requires to update the public parameters $f_i(x)$ of the related SC SCi.

Computation time over GF(2^512) and GF(2^160)



Time inverse (time for performing inverse element computation)



4.1 Security Analysis:

In this section all possible attacks will be addressed and also to bear all security attacks against the proposed scheme in the subsection.

4.1.1 Prevention of collusive attacks:

In the first possible attack is considered that any lower security class can to access the private key of the upper SC using his/her public parameters and its private key. In simple words, can SCj access the secret key of SCi using his/her secret and public parameters? Where SCj<=SCi. Let's prove that the proposed scheme is more secure against the collusive attack using example. As shown in the fig (20) the SC_6 wants to reveals the secret key $R_{3\,1}$ of the SC_3 using his/her public and secret parameters. For example SC_6 wants to derive his predecessor of SC_3 secret key H (r_3 K_{ca}) to encrypt the information as shown in the fig (20).

$$P_3(x) = H(\prod (x - (R_{3\,1})) + r_3K_{ca})$$

$R_{31} = H(r_3K_{CA}G) \neq r_3$ *(where* SC_6 can not derive the r_3 of SC_3*)*

So in the proposed scheme the secret parameter r_3 of security class SC_3 is secured against the attack because the SC_6 cannot access the SC_3 secret key $r_3$and also much secure due to hashing is non reversible H ($r_3K_{ca}G$).

4.1.2 Prevention of equation attacks:

In this type of attacks the two or more SC(s) have the common successor. And any one of them wants to try, the public parameter of any other SC for deriving the unauthorized private keys. So this type of attack is not possible in the proposed scheme because successor's encoded keys are encrypted using the private keys of his predecessor. Shown in fig (20), SC_3 and SC_4 have the common successor SC_5. The SC_4 wants to derive the SC_6's encryption key by using SC_3 public polynomial $f(x_3)$. To declare that the proposed scheme is secure against the possible attack let's prove by example by using fig (20). The relationship is shown in the fig (20) that $SC_3 > SC_5$ and $SC_4 > SC_5$. SC_4 tries to derive the SC_6's encryption key $R_{6\,4}$ through their common successor SC_5. With the information of $f_5(x)$ and encryption key $R_{5\,4}$, the SC_4 cannot derive SC_6's encryption key $R_{6\,4}$. Here the $f_5(x)$ is constructed by the points $P_5(x) = H(\prod (x - (R_{5\,4}))\ (x - (R_{5\,3}))\ (x - (R_{5\,1})) + r_5K_{ca})$ and $P_6(x) = H(\prod (x - (R_{6\,3}))\ (x - (R_{6\,1})) + r_6K_{ca})$, where $R_{6\,3} = H\ (R_6\ Y_{ca})$ and $R_{5\,3} = H\ (R_5\ Y_{ca})$, and also it is hard to compute for SC_4 to decrypt the SC_6's security key$R_{6\,4}$ from the public polynomial of and SC_5's secret key $R_{5\,4}$ without knowing security class SC_3's secret key $R_{3\,4}$ and also all the secret keys is applied the hash function which is more secured so this type of attack cannot break the secret keys of SC(s).

4.1.3 Prevention of exterior root finding attacks:

Consider the situation that attacker who is not a user in the hierarchy wants to atempt the encryption key of *SC* through the root finding algorigthm. Every successor's encryption keys of a SC_i are well secured into SC_i. Central authority CA update the public parameters $f'_i(x)$. In spite of that, for those successr's, which remain as successor's of SC_i in $f'_i(x)$, their secret are up to now at the same positions of $f'_i(x)$. the attackers able to produce a polynomial by taking the difference of $f_i(x)$ and $f'_i(x)$. So the attacker may refer *(Ben-Or,1981; Cohen, 1991)* to roots of the equation $f_i(x) - f'_i(x) = 0$ in a polynomial time. having the information of the roots, the attackers easily achieve the private keys of the successor's of SCi. In the proposed scheme if the attacker can calculate the x-coordinates from the equation $f_i(x) - f'_i(x) = 0$. So the attacker have to get $K_i = H\ (R_i Y_{ca})$ for $SC_i > SC_j$. Therefore it is infeasible to compute the R_{ij} of SC_i. And also the R_{ij} is encrypted by the key $H\ (R_i Y_{ca})$, so proposed scheme is secure against such type of attack.

4.1.4 Group confidentiality:

In group confidentiality attacker have no access to any secret key that can decode any information in the user hierarchy. The proposed scheme all the information about SC is encrypted through his secret keys, from the above security analysis the security of the private keys is secured using hash function. The secret informaion about all the SC security classes can access that inforamtion using its own secret keys. The attackers will face the reversibility of the hash function. So therefore the proposed scheme is protectd against the group confidentiality.

4.1.5 Forward and back backward confidentiality:

In the forward confidentiality the security classes delete from the hierarchy not able to access any sensitive information. Backward confidentiality is that security classes add into the hierarchy cannot access any secret information. The proposed scheme thinks about the dynamic key management and so it can gain the forward and backward confidentiality if updating the private key is performed under above two conditions.

Chapter 5

Conclusion and Future Work

5. Conclusion and future work.

The proposed scheme is An efficient & secure key management scheme for dynamic hierarchical access control, where ECC is used by [19-26]and the ECC gives high security and more efficient performance than the first generation public key techniques (RSA and Diffie-Hellman) now in use. As vendors look to upgrade their systems they should seriously consider the EC alternative for the calculation and bandwidth benefits they offer at comparable security. ECC uses a relatively small encoded key, a value that must be fed into the encryption algorithm to decrypt an encrypt message. This small key is faster and needs less calculation power than other first-generation encryption public key algorithms. E.g., a 160-bit ECC encryption key gives the similar security as a 1024-bit RSA encryption key and can be up to 15 times faster, depending on the platform on which it is implemented. RSA is a first-generation public-key cryptography method invented by Ronald Rivest, Adi Shamir and Leonard Adleman in the late 70s. Both RSA and ECC are in widespread use. The benefits of ECC over RSA are particularly important in wireless devices, where calculating power, memory and battery life are limited. So in this research paper shows clearly that there is some security problem in the previous published paper e.g. in Jeng-Weng [6] is vulnerable against exterior root finding attack. Also Chung et al.'s [9] also have some security leak like when updation in the hierarchy or add or delete some security class. In Nikooghadam, et al., [10] and Wu et al.'s schemes [15] require large storage apace in the public domain and require to update a hige number of private keys and size of the hierarchy is huge and at the end the dynamic access problem will become slow. The proposed scheme is an efficient & secure key management scheme for dynamic hierarchical access control based on ECC, Compared with the recently proposed schemes, the proposed scheme is secure against all the security attacks also the compromising attack and exterior root finding attack. Compared with the recently methods, both time complexity and storage space are considerably reduced in the proposed scheme. The proposed scheme achieved the higher security with low computational cost and storage cost with the help of smaller key sizes as compared to other schemes. Furthermore the proposed scheme is an efficient and secure against all the security attacks.

References

References:

1. Akl, S, G Tayllor, P.D. "Cryptographic solution to a problem of access control in a hierarchy," *ACM Transaction on Computer System*, vol. 1, pp. 239-247, 1983.
2. Mackinnon, S.J., Taylor, P.D Meijer, H., Akl S.G., "An optimal algorithm for assigning cryptographic keys to control access in a hierarchy," *IEEE Transactions on Computer*, vol. 34, pp. 797-802, 1985.
3. Harn, L., Lin, H.Y.,"A cryptographic key generation scheme for multilevel data security," *Computer and Security*, vol. 9, pp. 539-546, 1990.
4. Chang, C.C., Hwang, R.J., Wu, T.C., "Cryptographic key assignment scheme for access control in a hierarchy," *Computer and Mathematics with Application*, vol. 26, pp. 71-76, 1992.
5. Hsu, C.L., Tsai, P.L., Chou, Y.C., "Robust dynamic access control scheme in a user hierarchy based on one-way hash function," *The Journal of System and Software*, vol. 67, pp. 99-107, 2008.
6. Jeng, F.G., Wang, C.M., "An efficient key management scheme for hierarchal access control based on elliptic curve cryptosystem," *The Journal of System and Software*, vol. 79, pp. 1161-1167, 2006.
7. Shen, V.R.L., Chen, T.S., "A novel key management scheme based on discrete logarithms and polynomial interpolations," *Computers and Software*, vol. 21, pp. 164-171, 2002.
8. Yang, C., Li, C., "Access control in a hierarchy using one-way hash functions," *Computers and Security*, vol. 23, pp. 659-664, 2004.
9. Y. L. Lin, C. L. Hsu. "Secure key management scheme for dynamic hierarchical access control based on ECC," *The Journal of System and Software*, vol. 84, pp. 679- 685, 2011.
10. Nikooghadam, M., Zakerolhosseini, A., Moghaddam, M.E. "Efficient utilization of elliptic curve cryptosystem for hierarchal access control," *The Journal of system and software*, vol. 83, pp. 1917-1929, 2010.

11. Lin, Y., Hue, C., Wu, T., Yen, S., Tseng, C. "Secure key management scheme for hierarchical access control based on ECC," *Conference publications*, vol. 43, pp. 335-338, 2009.
12. Ben-Or, M, "Probabilistic algorithm in finite fields," *Foundation of computer sciences*, vol. 22, pp. 394-398, 1981.
13. Johnson, D., Menezes, A., Vanstone, S., "The Elliptic curve digital signature algorithm (ECDSA)", *International Journal of Information Security*, vol. 1(1), pp. 36-63, 2001.
14. Odelu, A., Das, A., Goswami, A. "An Efficient and secure key-management scheme for hierarchical access control in e-medicine system," *Springer*, vol.37, pp. 2325-37, 2013.
15. Basu, A., Sengupta, I., Sing, J.K., "Secured Cloud Storage Scheme Using ECC Based Key Management in User Hierarchy," *Springer*, vol. 7093, pp. 175-189, 2011.
16. Wu, S., Chen, K., "An Efficient Key-Management Scheme for Hierarchical Access Control in E-Medicine System," *Springer*, vol. 36, pp. 2325-2337, 2012.
17. Chen, T. S., Huang, J. Y., "A novel key management scheme for dynamic access control in a user hierarchy," *Elsevier*, vol. 165, pp. 339-351, 2005.
18. Lin, Y. L., Hsu, C. L.,"secure key management scheme for dynamic hierarchical access control based on ECC," The *Journal of System and Software*, vol. 84, pp.679-685, 2011.
19. Chung, Y. F., Lee, H. H., Lai, F., and Chen, T.S., "Access control in user hierarchy based on ECC," *Inform. Sci*, vol. 178(1), pp 230-343, 2008.
20. Das, A. K., N.R., and Tripathy, L., "Cryptanalysis and improvement of an access control in user hierarchy based on elliptic curve cryptosystem," *Inform. Sci,* vol. 209, pp 80-92, 2012.
21. Koblitz, N., Menezes, A., and Vanstone, S.A., "The state of elliptic curve cryptography," *Des. Codes Crypto,* vol 19, pp 173-193, 2000.
22. Liao, H. Z., and Shen, Y. Y., "On the elliptic curve digital signature algorithm," *Tunghai Sci.* vol. 8, pp 109-126, 2006.
23. Wu, S., and Chen, K., "An efficient key management scheme for hierarchical access control in E-Medicine system," *J. Medicine System*, vol. 10, 2011.
24. Vanstone, S. A., "Elliptic curve cryptosystem the answer to strong, fast public key cryptography for securing constrained environment," *Information security technical report,* vol. 12, pp 78-87, 1997.

25. Atallah M. J., Blanton M., Fazio N., Frikken K. B., "Dynamic and efficient key management for access hierarchies," *Information system,* vol. 12, pp 1-43, 2009.

26. Shuhua W. Chen K., "An efficient key management scheme for hierarchical access control in e-medicine system," *Journal of Medical system, Springer,* vol. 10, 2011.

27. Wu, Z.-Y., Chung, Y., Lai, F., and Chen, T.-S., *"A password-based user authentication scheme for the integrated EPR information system."* J. Med. Syst. Vol. 36(2,), pp 31–638, 2012.

28. Odelu, A., Das, A., Goswami, A. *"An Effective and secure key-management scheme for hierarchical access control in e-medicine system."* Journal of Medical system, Springer, vol. 37, pp 1- 18, 2013.