Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 10, Number 5 (2014), pp. 663-684

http://www.ripublication.com

Integrating Symmetric Polynomials and Tree Based Group Elliptic Curve Diffie Hellman Scheme for providing Hierarchical Access Control in Government Organizations

Jeddy Nafeesa Begum[1*], Dr. Krishnan Kumar[2] and Dr. Vembu Sumathy[2]

[1]Department of Computer Science and Engineering, Government College of Engineering, Bargur Krishnagiri District , Tamil Nadu, India , Pin Code 635 104*
nafeesa.research@gmail.com
[2]Government College of Technology, Coimbatore, Tamil Nadu, India

Abstract

Hierarchical Access Control (HAC) refers to a control policy commonly used in organizations which allows members of staff belonging to a senior classes to access the messages which are transmitted among the members of staff belonging to subordinate classes whereas vice versa is not allowed. This paper deals with implementing the HAC by integrating Symmetric polynomials and Tree Based Group Elliptic Curve Diffie-Hellman (TGECDH) scheme. A scalable dynamic scheme can be achieved only if the implementation for hierarchical access control and dynamic group management are done in two different layers. The symmetric polynomial is a polynomial which gives a same value for different combinations of parameters. This property is exploited for deriving a scheme for providing HAC. A contributory key agreement scheme is used to provide forward and backward secrecy so that only active members of a class at any instant are able to retrieve the messages. The TGECDH scheme is used for managing the dynamic groups. A Trusted Intermediary Software Agent (TISA) is used to perform the dual encryption which facilitates a dynamic scalable hierarchical access control in the organization. The trouble of sending all keys of junior classes to all users of ancestor classes is solved by following this modular layered approach. It is found that the hierarchical access control is achieved with less communication, computation cost and storage cost.

Keywords: Hierarchical Access Control, Symmetric Polynomials, Elliptic Curve, Diffie Hellman, Software Agent, Forward Secrecy, Backward Secrecy.

Introduction

Just as organizations have goals describing their primary business objectives, they also have goals with respect to controlling how these objectives are met. These are the control goals of an organization which are imposed through a system of internal controls. Such a system enables them to adhere to external laws and internal regulations, prevent and detect fraud and continuously enhance the overall quality of the business. Independent of the type of organization, these internal control systems use common underlying principles to establish and achieve control over business activities. One such control is the Hierarchical Access Control.

Key establishment is the first step to develop all the other security mechanisms, because most security protocols depend on keys to operate correctly and provide desirable security performance. In this paper, a scalable protocol using symmetric polynomial and Diffie-Hellman scheme is introduced to provide Hierarchical Access Control. The proposed model is called as DEHAC (Dual Encryption Hierarchical Access Control) as the HAC is achieved by a process of Dual Encryption. It reduces the rekeying cost and hence the total cost involving the communication cost, computation cost and memory cost.

Literature review

The first cryptographic Hierarchical Access Control solution for enforcing access control policy was given by Akl and Taylor [1]. Unfortunately, one limitation is its complex operation for dynamic reconfiguration when nodes are added or removed from the hierarchy. Forward secrecy and backward secrecy are essential during membership changes. Concepts of backward secrecy and forward secrecy are introduced in secure group communication systems by X. Zou et. al [2]. Many schemes, like the one given by Sandhu [3], have been developed for providing HAC. The difficulty in the schemes is that both user dynamics as well as group dynamics cannot be ensured at the same time.

Atallah et. al [4] suggested a scheme in which the hierarchy is modeled as a set of partially ordered classes and a user who obtains access to a certain class can also obtain access to all descendant classes of her class through key derivation . Updates are handled locally but still maintaining the forward and backward secrecy is difficult. Symmetric polynomials have been used for HAC by Blundo et al [5] , Das et al [6] and X. Zou and L. Bai [7]. The symmetric polynomials have an important property that for different permutation of the variables, the value of the polynomial does not change. This property will help in deriving the key of the descendant classes. However the problem should be approached in Divide and Conquer technique. In [7], the authors have derived the descendant class key by using set theory operations on ancestral sets but the scheme fails to perform better for forward secrecy.

The best schemes for forming a key in a highly dynamic environment were put forth by W. Diffie and M. E. Hellman [8]. There are some variations on the scheme such as TGDH as given by Kim et. al [9] which are been found to be very efficient . Aparna et.al [10] and Yong Wang et.al [11] have discussed the advantages and disadvantages of the various key management schemes. Kumar et. al [12] have used

the region based approach for secure group communication . This instigated the thinking to use the symmetric polynomial approach for hierarchical access control and the Diffie-Hellman scheme to solve the problem of Dynamic Groups and provide an efficient scalable dynamic HAC Scheme.

Motivation

The hierarchical access control problem is more difficult to solve than a mere secure group communication problem because in addition to providing secure communication among users who are dynamic , it should ensure that ancestor class users active at a time should be able to see the messages which are transmitted between the respective descendant class users. The central idea which was visualized is that efficient HAC can be achieved by

i. Transmission of lesser number of encrypted messages.
ii. Restricting the transmission of keys.
iii. The encrypted message should not be the same in all parts of the network which means that in each and every local network, the local key should be used for encryption rather than a global key.
iv. Descendant keys should be derived or calculated by the ancestor classes rather than transmitted.
v. Use of automated agent referred as TISA in this scheme that acts as an intermediary in each security class to ensure that users belonging to higher class are able to perceive the messages that are transmitted by the users of the lower class by performing the dual encryption and decryption of the message.
vi. Consideration to use a key agreement scheme for the local groups which is best suitable for their infrastructure.
vii. Ability of every security class to change its own key independently.
viii. The authentication of users by performing security check.
ix. Implementing divide and conquer approach.

Dual encryption model for the proposed HAC solution

The Dual Encryption Hierarchical Access Control consists of i). HAC layer using symmetric polynomials ii). Dynamics group layer using TGECDH Scheme separately in each and every security class iii). Communication semantics for TISA for combining the two levels .The advantage of using the distributive divide and conquer approach is that it is less encryption intensive than the other approaches, since data is only re-encrypted at the security class affected by the membership change. However, other schemes require re-distributing the updated key to all the classes requiring the new key. The use of divide and conquer approach with the symmetric polynomials approach avoids distribution of keys and keys are derived rather than transmitted.

Two keys are used in each class. For convenience, the keys are denoted as CK_x and SP_x where $1 < x < n$ and n is the maximum number of security classes in the access control hierarchy. Each class is associated with two keys CK_x and SP_x .

1. Symmetric Polynomial Key(SP_x) is used to encrypt and decrypt the messages that are broadcasts among the Software Agents of the entire system.

2. The Class Key (CK_x) is used to encrypt and decrypt the messages that are broadcast within the members of the security class.

The $CK_x$of a particular security class is formed by the contributory key agreement scheme used in the security class. This satisfies the requirement that the data remains secret from all unauthorized users of the security class.

The following example is used to demonstrate the working of the proposed scheme. The hierarchy is followed in all government departments to ensure that the schemes implemented by the government reaches the needy people The District Administrations also have additional units for the planning , collaboration and networking the various units under their jurisdiction as shown in Figure 1.

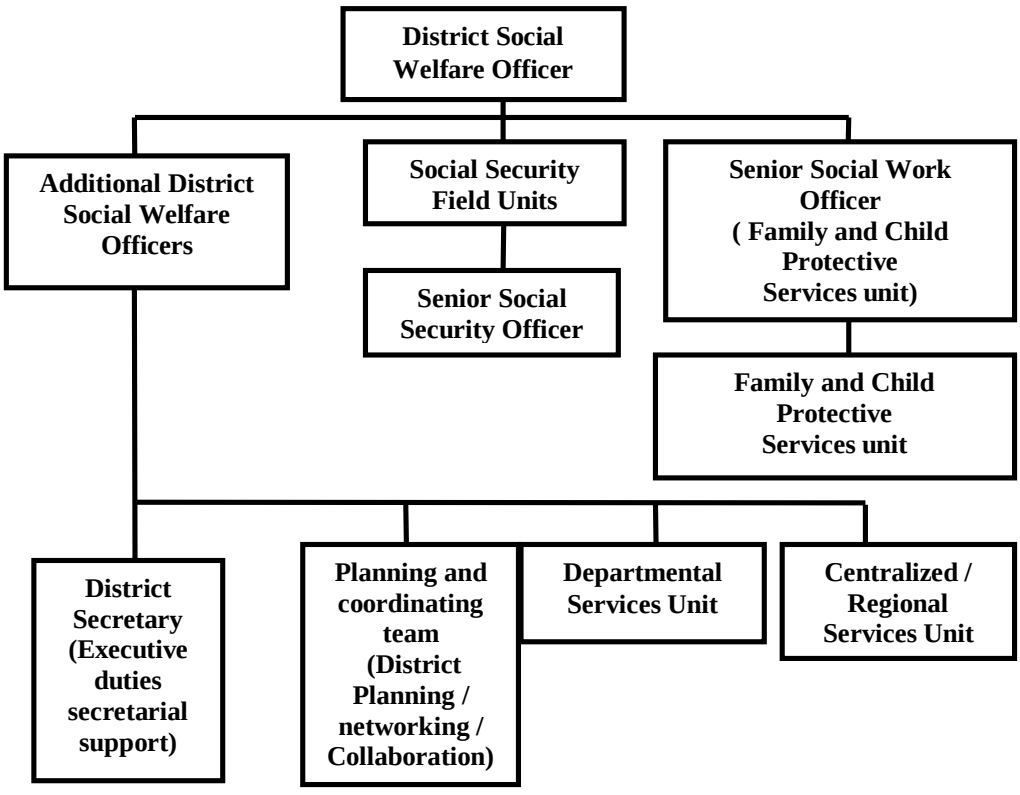

Figure 1: Hierarchy in a Government Department

The Proposed DEHAC scheme will be useful for transmitting the information about activities in the lower classes automatically to all the higher officials. The Hierarchy given in Figure 1 is modeled as shown in Figure 2.

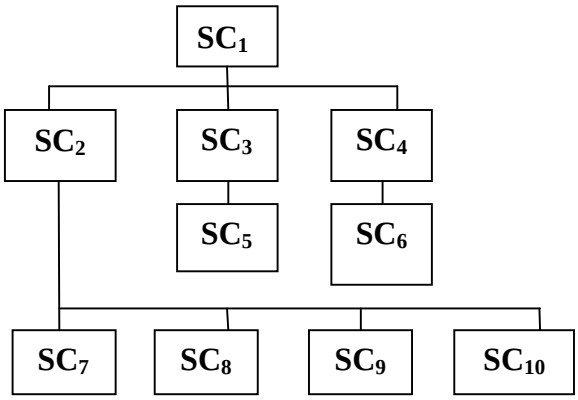

Figure 2: Modeled Hierarchy of a Government Department

The privileges attained by each and every security are shown in the Table 1 .

Layer-1: IMPLEMENTING THE HAC

The symmetric polynomial scheme is used in the design of layer-1. It consists of the following stages

- Polynomial distribution by the Central authority to TISA.
- Key calculation of the respective security Classes by TISA.
- Key derivation of descendant class TISA's by the ancestral Class TISA's.

The basics of symmetric polynomials are discussed in the following section before discussing the intricacies of the layer-1 design.

Table 1: Permissible Privileges of Security Classes

Class	Direct Privileges	Indirect Privileges	Effective Privileges
SC_1	SC_1	SC_2,SC_3,SC_4,SC_5,SC_6,SC_7,SC_8,SC_9 , SC_{10}	SC_1,SC_2,SC_3,SC_4,SC_5,SC_6, SC_7,SC_8,SC_9,SC_{10}
SC_2	SC_2	SC_7,SC_8,SC_9 , SC_{10}	SC_2,SC_7,SC_8,SC_9 , SC_{10}
SC_3	SC_3	SC_5	SC_3,SC_5
SC_4	SC_4	Nil	SC_4
SC_5	SC_5	Nil	SC_5
SC_6	SC_6	Nil	SC_6
SC_7	SC_7	Nil	SC_7
SC_8	SC_8	Nil	SC_8
SC_9	SC_9	Nil	SC_9
SC_{10}	SC_{10}	Nil	SC_{10}

Symmetric Polynomials

Symmetric polynomials have played a key role in many areas of mathematics including the theory of polynomial equations, representation theory of finite group, mathematical physics , quantum mechanics and solutions. In the proposed scheme the symmetric polynomials are used for providing HAC.

Definition 1: Polynomial

A polynomial in a single indeterminate can be written in the form $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_2 x^2 + a_1 x + a_0$ . . A polynomial in n determinates can be written in a form

$$g(x_{1,\ldots}, x_n) = \sum_{i_1,\ldots i_n} a_{i_1 \ldots i_n} \quad \mathrm{x}_1^{i_1} \cdots \mathrm{x}_\mathrm{n}^{\mathrm{i_n}} \tag{1}$$

Where the a's are elements and the exponents are non- negative integers.

Definition 2: Symmetric Polynomials

A polynomial $g(x_{1,\ldots}, x_n)$ is symmetric if for any permutation τ of $\{1, \cdots n\}$, $g(x_{\tau(1)}, \cdots, x_{\tau(n)}) = g(x_1, \cdots, x_n)$.

Definition 3: Elementary Symmetric Polynomials

Let $x_{1,\ldots} x_n$ denote indeterminates. The elementary symmetric functions in $x_{1,\ldots}, x_n$ are the polynomials σ_i given by sums of all products of different $x_{j'}s$ [13]

$\sigma_1 = x_1 + x_2 + \cdots x_n$

$\sigma_2 = x_1x_2 + x_1x_3 + \cdots + x_2x_3 + \cdots + x_{n-1}x_n$

$\vdots$

$\sigma_n = x_1x_2 \cdots x_n$

Definition 4: Complete Symmetric Polynomials

The k[th] complete symmetric polynomial $h_k(z_1, z_2, \cdots, z_n)$ on the n variables $\{z_1, z_2, \cdots, z_n\}$ is the sum of all possible products of k of these variables chosen with replacement [14].

The first few elementary and complete symmetric polynomials on two variables x, y is shown in Table 2 and three variables x, y, z is shown in Table 3.

Table 2: Symmetric Polynomials on Two Variables.

k	$e_k(x,y)$	$h_k(x,y)$
0	1	1
1	$x + y$	$x + y$
2	xy	$xy + x^2 + y^2$

Table 3: Symmetric Polynomials on Three Variables.

k	$e_k(x,y,z)$	$h_k(x,y,z)$
0	1	1
1	$x+y+z$	$x+y+z$
2	$xy+xz+yz$	$xy+yz+zx+x^2+y^2+z^2$
3	xyz	$x^3+y^3+z^3+x^2y+x^2z+y^2x+y^2z+z^2x+z^2y+xyz$

Polynomial distribution by the central authority to TISA

Before deployment, a global polynomial pool G of multivariate polynomials is kept by the Central Authority. Each polynomial has a unique polynomial ID. CA randomly generates a symmetric polynomial in m variables. The value of m indicates the maximum level allowed in the hierarchy. The polynomial function $P(x_1, x_2, \cdots x_m)$ is kept as secret by the CA. Every TISA of a security class in the hierarchy has a polynomial function which is derived from $P(x_1, x_2, \cdots x_m)$ and the polynomial function is transmitted to each TISA securely by the Central Authority. The Central authority uses some global numbers i.e. n random numbers s_i associated with SC_i for i = 1, 2 ,…n and (m - 1) additional random numbers $s_j^{'}$ for j = 1,2,… m - 1 . $s_i$and s_j belong to Z_p.

Key calculation of the respective security classes by TISA.

The $TISA_i$ calculates the key SP_i using the following formula given in (2)

$$SP_i = g_i\left(s_{1,}^{'} s_{2,}^{'} \dots\dots, s_{m-m_i-1,}^{'}\right) = P(s_{i,}\, s_{i1}, s_{i2,\dots}, s_{im_i}\, s_{1,}^{'} s_{2,}^{'} \dots\dots, s_{m-m_i-1}^{'}) \quad (2)$$

This key can be changed by the central authority during Class Dynamics. The key is known only to $TISA_i$.

Key derivation of descendant class TISA's by the ancestral Class TISA's.

The main issue concerned with HAC is that the TISA of the ancestor Class (SC_i) should be able to derive the key of the descendant Class(SC_j). In the key derivation , a new ancestral Set $S_{j\backslash i}$ that is used to identify the collection of ancestors for Class SC_j but excluding SC_i and those classes which are ancestors of both classes SC_i and SC_j .

$$S_{j/i} \triangleq S_j(S_i \cup \{SC_i\}) = \left\{SC_{(j\backslash i)_1}, SC_{(j\backslash i)_2}, SC_{(j\backslash i)_{r_j}}\right\} \quad (3)$$

Where $r_j = \left|S_{j/i}\right|$ and $(j/i)_l$ is an ordinal number $1 \le (j/i)_l \le n$ $1 \le$ (j/i) $_l$ for $l = 1,2,\cdots, r_j$.

The descendant class key SP_j can be calculated by using the formula given in Equation 4 and Equation 5.

$$SP_j = g_i(s_j, s_{(j\backslash i)_1}, s_{(j\backslash i)_2}, \dots, s_{(j\backslash i)_{r_j}}, s_{1}^{'}, s_{2,}^{'} \dots, s_{m-m_{i-2-r_j}}^{'}) \quad (4)$$

$$=P(s_{i,}s_j, s_{i1}, s_{i2}, \cdots, s_{im_i}, s_{(j\backslash i)_1}, s_{(j\backslash i)_2}, \cdots, s_{(j\backslash i)_{r_j}}, s_1^{'}, s_{2,}^{'} \cdots, s_{m-m_{i-2-r_j}}^{'}) \quad (5)$$

as s_i and $s_j^{'}$ are globally known, the TISA of the class SC_i can compute its key and the descendant class key but not its ancestors' key using the polynomial function assigned to class SC_i using the function $g_i(x_{mi+2}, x_{mi+3}, \cdots, x_m)$If it tries to calculate the ancestor class key, either the polynomial will give an incorrect key or the function will fail due to mismatch in parameters.

For the modeled hierarchy shown in Figure 2, the following are discussed

1. Calculation of their own symmetric polynomial key by the respective TISA 's of the security classes.
2. Derivation of symmetric polynomial key of the descendant classes by all the ancestor classes.
3. The failure of non ancestor classes to derive the key of any particular class.

The set of Classes = { SC_1, SC_2, SC_3, SC_4, SC_5, SC_6,SC_7,SC_8,SC_9,SC_{10}).
Set of Ancestor Classes = { SC_1, SC_2, SC_3 , SC_4}. The TISA calculates the symmetric polynomial key on the behalf of each and every class as shown below.
The $TISA_1$, $TISA_2$, … $TISA_{10}$ calculate the symmetric polynomial key SP_1,SP_2,…SP_{10}. To enable the calculation and derivation of symmetric polynomial keys, a value m which identifies the level of hierarchy that may be supported is to be calculated. The value of m should be greater than or equal to max(m_1,m_2,m_3,m_4,m_5,m_6) + 1.
m_1= number of ancestor classes for the security class SC_1 = { Φ} = 0.
m_2 = number of ancestor classes for the security class SC_2 = { SC_1} = 1
m_3= number of ancestor classes for the security class SC_3 = { SC_1} = 1
m_4= number of ancestor classes for the security class SC_4 = { SC_1} = 1
m_5= number of ancestor classes for the security class SC_5 = { SC_1 ,SC_3} = 2
m_6= number of ancestor classes for the security class SC_6 = { SC_1 ,SC_4} = 2
m_7= number of ancestor classes for the security class SC_7 = { SC_1 ,SC_2} = 2
m_8= number of ancestor classes for the security class SC_8 = { SC_1 ,SC_2} = 2
m_9= number of ancestor classes for the security class SC_9 = { SC_1 ,SC_2} = 2
m_{10}= number of ancestor classes for the security class SC_{10} = { SC_1 ,SC_2} = 2
so m ≥ max (0 , 1 , 1 , 2 , 2 , 2) +1 = 3 . The m value of 3 is suffice for the hierarchy shown above , however a large value of m makes class dynamics easier when more security classes need to be added . Here m is chosen as 4 so that up to 4 levels of ancestors can be used in the hierarchy. The parameters for the hierarchy are s_1,s_2,s_3,s_4 , s'_1, s'_2, s'_3.

The Central Authority randomly generates a polynomial function $P(x_1, x_2, x_3, x_4)$ with four parameters. The CA can then compute ten polynomial functions for classes SC_i. Once the polynomial functions are obtained, they are securely transmitted to every TISA respectively. The calculation of the symmetric polynomial key by the TISA of the respective classes is shown below.

Calculation of SP_1 by TISA $_1$
H_1 = Ancestor Classes of SC_1 = { Φ }
$m_1 = 0$, m- m_i -1 =5
$SP_1 = g_1(s'_1, s'_2, \ldots, s'_{m-mi-1}) = P(s_i, s_{i1}, s_{i2}, \ldots, s_{im}, \ldots, s'_1, s'_2, \ldots, s'_{m-mi-1})$
$SP_1 = g_1(s'_1, s'_2, s'_3) = P(s_1, s'_1, s'_2, s'_3)$

Calculation of $SP_2$by TISA $_2$
H_2 = Ancestor Classes of SC_2 = { SC_1}
$m_2 = 1$, m- m_i -1 =2
$SP_2 = g_2(s'_1, s'_2, \ldots, s'_{m-mi-1}) = P(s_{i,}, s_{i1}, s_{i2}, \ldots, s_{im}, \ldots, s'_1, s'_2, \ldots, s'_{m-mi-1})$
$SP_2 = g_2(s'_1, s'_2) = P(s_2, s_1, s'_1, s'_2)$

Calculation of SP_3 by TISA $_3$
H_3 = Ancestor Classes of SC_3 = { SC_1}
$m_1 = 1$, m- m_i -1 =2
$SP_3 = g_3(s'_1, s'_2, \ldots, s'_{m-mi-1}) = P(s_i, s_{i1}, s_{i2}, \ldots, s_{im}, \ldots, s'_1, s'_2, \ldots, s'_{m-mi-1})$
$SP_3 = g_3(s'_1, s'_2) = P(s_3, s_1, s'_1, s'_2)$

Calculation of SP_4 by TISA $_4$
H_3 = Ancestor Classes of SC_4 = { SC_1}
$m_1 = 1$, m- m_i -1 =2
$SP_4 = g_4(s'_1, s'_2, \ldots, s'_{m-mi-1}) = P(s_i, s_{i1}, s_{i2}, \ldots, s_{im}, \ldots, s'_1, s'_2, \ldots, s'_{m-mi-1})$
$SP_4 = g_4(s'_1, s'_2) = P(s_4, s_1, s'_1, s'_2)$

There are four ancestors in the hierarchy. SC_1 is the ancestor of SC_2 ,SC_3,… SC_{10} . SC_2 is the ancestor of SC_7, … SC_{10}. SC_3 is the ancestor of SC_5 and SC_4 is the ancestor of SC_6. Key derivation of the subordinate classes is done by the ancestor classes using the equations 3 ,4 and 5 which includes the following steps

i) Consider the security class as j for which the key is derived (subordinate class)
   S_j = ancestral set of node j.
ii) Consider the Security Class as i, which derives the key (ancestor class)
   S_i=ancestral set of node i
iii) Calculate H_i U{ SC_i}
iv) Calculate r_j = | $H_{j/i}$ | [set subtraction]
v) Key derivation formula given in equation is used
vi) The notation $AC_{i,j}$ means that ancestor class SC_i derives the key of the descendant class SC_j.

Some examples of key derivations by the ancestral classes are as follows

a) Key derivation of SC_2 by SC_1
j = 2 i =1
$H_j/_i = H_2/ H_1$ U SC_1
= { SC_1} /{ Φ U SC_1}

$= 0$
$r_j = 0$
$m- m_i-2-r_j = 4 - 0- 2-0=2$
$AC_{1,2}= P(s_1, s_2, s_1', s_2')$
$SP_2= P(s_2, s_1, s'_1, s'_2)$
$AC_{1,2} = SP_2$ (same parameters in different permutation)

b) Key derivation of SC_4 by SC_1
$j = 4$ $i =2$
$H_j /_i = H_4/ H_2 U SC_2$
$= \{ SC_1\} /\{ \Phi U SC_1\}$
$= 0$
$r_j = 0$
$m- m_i-2-r_j = 4 - 0- 2-0=2$
$AC_{1,4}= P(s_1, s_4, s_1', s_2')$
$SP_4=P (s_4,s_1,s'_1, s'_2)$
$AC_{1,4} = SP_4$ (same parameters in different permutation)

The following cases shown in Table 4 are non permissible privileges for the example hierarchy in Figure 1 . When a class which is not an ancestor tries to derive the key it results in a polynomial with different parameters or mismatch in parameters thereby generating a wrong key.

Table 4: Non permissible Privileges

Class	Non Permissible Privileges	Class	Non Permissible Privileges
SC_1	-	SC_6	$SC_1,SC_2,SC_3,SC_4,SC_5,SC_7$ SC_8,SC_9,SC_{10}
SC_2	$SC_1,SC_3,SC_4, SC_5,SC_6$	SC_7	$SC_1,SC_2,SC_3,SC_4,SC_5,SC_6,$ SC_8,SC_9,SC_{10}
SC_3	SC_1,SC_2,SC_4,SC_6,SC_7 SC_8,SC_9,SC_{10}	SC_8	$SC_1,SC_2,SC_3,SC_4,SC_5,SC_6,SC_7$, SC_9,SC_{10}
SC_4	SC_1,SC_2,SC_3,SC_5,SC_7 SC_8,SC_9,SC_{10}	SC_9	$SC_1,SC_2,SC_3,SC_4,SC_5,SC_6,SC_7$ SC_8,SC_{10}
SC_5	$SC_1,SC_2,SC_3,SC_4,SC_6,SC_7$ SC_8,SC_9,SC_{10}	SC_{10}	$SC_1,SC_2,SC_3,SC_4,SC_5,SC_6,SC_7$ SC_8,SC_9

A few Examples for the key derivation by the non-ancestral classes shown in Table 4 is discussed below

Case: 1
Key derivation of SC_1 by SC_2
$j = 1$ $i =2$
$H_j /_i = H_j/ H_i U SC_i$

$= \{ \Phi \} / \{ SC_1 \cup SC_2 \} = \{ \Phi \}$

$r_j = 0$

$m - m_i - 2 - r_j = 6 - 1 - 2 - 0 = 1$

$NAC_{2,1} = P(s_2, s_1, s_1, s_1')$

$SP_1 = (s_1, s_1', s_2', s_3')$

$NAC_{2,1} \neq SP_1$ (parameters are not correct hence Security Class SC_2 does not get the correct key of Security Class SC_1)

Case : 2

Key derivation of SC_4 by SC_3

$j = 4$ $i = 3$

$H_j / _i = H_4 / H_3 \cup SC_3$

$= \{ SC_1 \} / \{ SC_1 \cup SC_3 \} = \{ \Phi \}$

$r_j = 0$

$m - m_i - 2 - r_j = 4 - 1 - 2 - 0 = 1$

$NAC_{3,4} = P(s_3, s_4, s_1, s_1')$

$SP_4 = P (s_4, s_1, s'_1, s'_2)$

$NAC_{3,4} \neq SP_4$ (parameters are not correct hence Security Class SC_3 does not get the correct key of Security Class SC_4)

Layer-2: The formation of Contributory key using TGECDH

Tree based Group Elliptic Curve Diffie-Hellman (TGECDH) protocol is used for maintaining the key in each class .This example shows how the shared key is obtained by the members of a class. The same operations happen in each and every security class for formation of their respective local key CK. In the class, initially two members M_1 & M_2 are available. If a new member M_3 wants to join the class, it broadcasts a join request message to class controller. The class controller receives this message and determines the insertion point in the tree. If a member joins in the shallowest rightmost node there, it does not increase the height of the key tree. If the key tree is fully balanced, the new member joins the root node. The controller is the rightmost leaf in the sub tree rooted at the insertion node. When a member joins in the class, it creates a new node and promotes the new node to be the parent of both the insertion node and the new member node. After updating tree, the class controller proceeds to update its share and passes all public keys tree structure to new member.

The new member acts as the new class controller and computes the new class key. Next, the class controller broadcasts the new tree that contains all public keys. All other members update their trees accordingly and compute the new class key.

If a member wants to leave the class, first it should send the leave request to the class controller to generate the new key. When the leave request message is received by class controller, it updates its key tree by deleting the leaf node corresponding to leave member. The former sibling of leave member is promoted to parent node. The class controller generates a new private key share, computes all public key pairs on the key-path up to the root and broadcasts the new key tree that contains all public keys. The entire members in the class compute the new group key.

$BK_{<l,v>}$ = $K_{<l,v>}$* G.
$K_{<l,v>}$ = r_v* G.
Where
$K_{<l,v>}$, … private key
$BK_{<l,v>}$ … public key
r_v … random number
G … Generator

The intermediate node with two children does not represent any class but it represents a sub-class. The intermediate node's private key is treated as the sub-class key. It can be calculated by the following rule where node < l, v >'s two children are < l + 1, 2v > and < l+ 1, 2v + 1 > Where l is the level, v is the vertices index.

$K_{<l,v>}$ = $X_{co}(K_{<l+1,2v>}*BK_{<l+1,2v+1>})$
= $X_{co}(K_{<l+1,2v+1>}*BK_{<l+1,2v>})$
= $X_{co}(K_{<l+1,2v>}* K_{<l+1,2v+1>}* G)$

Where
X_{co} … is the x-coordinate of the point represented within the parentheses.
l … is the height (level) of the node and
v … is the index of the node at level l

The numerical illustration for the formation of the contributory key when M_3 Joins the security class SC_2 is explained
When a new member M_3 joins the group, the previous class controller M_2 changes its private key value from 14755 to 8751 and passes the public key tree to Member M_3. New private key is $K'_{<2,2>}$

$K'_{<2,2>}$ = 8751
$BK_{<2,2>}$ = $K'_{<2,2>}$ *G= (8751mod 769) *G = 292G
= 292 * (0,376) = (7,177)
$K_{<1,1>}$ = $X_{co}(K_{<2,2>}*BK_{<2,3>}) = X_{co}(9751*(725,224))$
= $X_{co}(9751*122G)= X_{co}(675,243) = 675.$
$BK_{<1,1>}$ = $K_{<1,1>}$ * G
= 675G= 675 * (0,376) = (127,150)

Now, M_2 becomes new controller. Then, M_3 generates the public key (725, 224) from its private key as 48569 and computes the group key as (641,685) shown in Figure 3.

$K_{<2,3>}$ = 48569
$BK_{<2,3>}$ = $K_{<2,3>}$*G
= (48569 mod 769) G= 122G
= 122 * (0,376) = (725, 224)
$K_{<1,1>}$ = $X_{co}(K_{<2,3>}*BK_{<2,2>}) = X_{co}(9751*(725,224))$
= $X_{co}(9751*122G) = X_{co}(675,243)= 675.$
$BK_{<1,1>}$ = $K_{<1,1>}$ * G
= 675G= 675* (0,376)= (127,150).

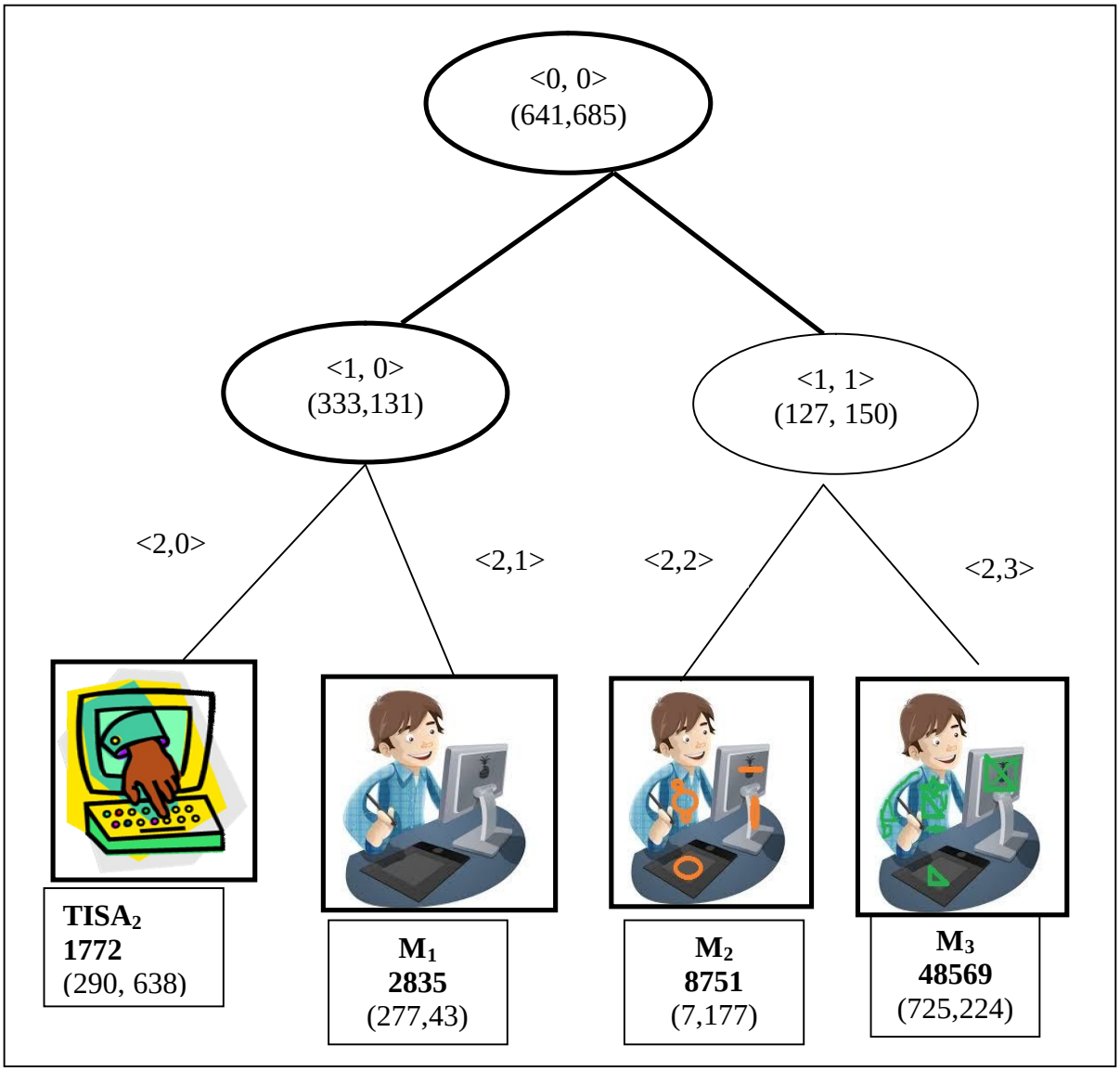

Figure 3 : User M_3 joins

The class key is computed as follows

$K_{<0,0>}$ = $X_{co}(K_{<1,1>} * BK_{<1,0>})$ = $X_{co}(675* (333,131))$

= $X_{co}(149*675G)$ = $X_{co}(355,103)= 355$

$BK_{<0,0>}$ = $K_{<0,0>} * G$

= $355G= 355* (0,376) = (641,685)$.

M_3 sends public key tree to all members. Now, Member M_1, M_2 compute their class key.

Member node <2,0> and <2,1>

$K_{<0,0>}$ = $X_{co}(K_{<1,0>} * BK_{<1,1>})= X_{co}(149*675G)$

= $X_{co}(605G)=$ $X_{co}(355,103)= 355$

$BK_{<0,0>}$ = $K_{<0,0>}$ * G
= 355G= 355* (0,376) = (641,685).
Member node <2,2 >
$K_{<0,0>}$ = $X_{co}(K_{<1,1>}*BK_{<1,0>})$ =X_{co}(675* (333,131))
= X_{co}(149*675G) = X_{co}(355,103)= 355
$BK_{<0,0>}$ = $K_{<0,0>}$ * G
= 355G= 355* (0,376) = (641,685).

Combining the two levels

The following section discusses the methodology by which the two levels work together to provide Scalable Dynamic Hierarchical Access Control by the convenience of the Communication Semantics.

Role of TISA in each Class:

In the proposed scheme, it is envisaged that there are TISA's in each and every security class. The TISA has the following functionality

- It participates in the contributory key agreement to form the local key of their class.
- It encrypts the messages by using the corresponding symmetric polynomial key of their class and broadcasts to the fellow TISA's associated with other classes.
- On receiving encrypted messages from the peer TISA's , the TISA decrypts the message after deriving the key of the descendant classes. It then encrypts the message with the local key and transmits to the users of its security class.

Communication Semantics for HAC

The communication semantics in the Upper Layer and Lower Layer is explained below

a. Communication Semantics within the Security Class

The sender member encrypts the message with the class key (CK) and multicasts it to all member in the security class. The security class members receive the encrypted message, perform the decryption using the class key (CK) and acquire the original message. The communication operation is as follows.

$$Sender \xrightarrow{E_{CK[message]\ \&\ Multicast}} Receiver$$

$$Receiver \xrightarrow{D_{CK}\left[E_{CK[message]}\right]} Original\ Message$$

b. Communication Semantics for Ancestor Classes

The sender of the message encrypts the message with the class - key (CK_x) and multicasts it to all the members in the security class, class controller ,TISA. The TISA decrypts the message with class key and encrypts with the Symmetric polynomial key (SP_x) and multicasts it to the Ancestor TISA 's . The Ancestor class

TISA's of all security classes first derive the key of the lower class symmetric polynomial key decrypt the message with derived key . They then encrypt with message with the respective class key and multicasts it to all the members in the security class. Each member in the security class receives the encrypted message and performs the decryption operation using class key and gets the original message. In this way the dual encryption model protocol performs communication. The communication semantics are as follows.

$$Sender \xrightarrow{E_{CK}[Message]\&\ Multicasts} TISA$$

$$TISA \xrightarrow{D_{CK}[E_{CK}[Message]]\&\ Multicasts} Original\ Message$$

$$TISA_{descendants} \xrightarrow{E_{SP}[Message]\&\ Multicast} TISA_{ansestor}$$

$$TISA_{anscestors} \xrightarrow{applies\ symmetric\ polynomial\ derivation} Symmetric\ Polynomial\ Key$$

$$TISA_{anscestors} \xrightarrow{D_{SP}[E_{SP}Message]]} Original\ Message$$

$$TISA_{ancestors} \xrightarrow{E_{CK}[Message]\&\ Multicast} Members\ of\ the\ Class$$

$$Members\ of\ Class_{(anscesor\ groups)} \xrightarrow{D_{CK}[[E_{CK}Message]]} Original\ message\ of\ the\ descendant\ users$$

Performance Analysis

Storage Cost

Memory cost is directly proportional to the number of members in case of TGDH and GDH. So, when the members go on increasing, TGDH and GDH occupy large memory space. In GDH each member needs memory space to store its private key, class key and n+ 1 additional public key. Members in TGDH have to store all keys in their key-paths and 2n-2 public keys. In TGDH, it depends on the level of the members. A new member in a deeper level needs to store more keys. Approximately $\lceil \log_2 n \rceil + 1$ keys are in the key-path.

Table 5 gives the formula used for calculating the storage cost of DEHAC using Symmetric Polynomial Based GDH Scheme, DEHAC using Symmetric Polynomial Based TGDH Scheme, and DEHAC using Symmetric Polynomial Based TGECDH Scheme and Table 6 gives key size equivalents for ECC based Schemes.

TABLE 5: Storage cost for the proposed schemes

S. No	Scheme Used	Level - 1 SP Scheme	Level- 2 CKA Scheme		Total Storage Cost
			Number of Private Keys	Number of Public Keys	
1	SP_GDH	$\binom{w+k-1}{w-1}$ coefficients + 2 x w random parameters	2	n+1	$\binom{w+k-1}{w-1} + 2w + 2 + n + 1$
2	SP_TDGH	$\binom{w+k-1}{w-1}$ coefficients + 2 x w random parameters	L+1	2n-2	$\binom{w+k-1}{w-1} + 2w + L + 1 + 2n - 2$
3	SP_TGECDH	$\binom{w+k-1}{w-1}$ coefficients+ 2 x w random parameters	L+1	2n-2	$\binom{w+k-1}{w-1} + 2w + L + 1 + 2n - 2$

Where L is the level of the member in the Tree , n is the number of members in the security class ,w is the number of variables in the symmetric polynomial and t is the threshold .

TABLE 6: Key size for equivalent security

Public Key			RSA Key Length for approximate equivalent security	Private key length for approximate equivalent security	Key size ratio
ECC Key Length					
Prime Field	Binary Field K_i	Public key Pk_u			
112	113	224	512	56	1:4
128	131	256	704	64	1:5
160	163	320	1024	80	1:6
192	193	384	1536	96	1:8
224	233	448	2048	112	1:9
256	283	512	3072	128	1:12
384	409	768	7680	192	1:20
521	571	1042	15360	256	1:29

Where K_i is Private key size in bits and PK_u Public key size in bits. Always public key size is twice that of private key in ECC. Figure 4 shows the graph for the bit storage for the three variations when number of users are 9,10,11,12 and Figure 5 for the number of users being 35,40,45,50 and 55. It is seen that the SP-TGECDH scheme occupies less storage compared to other schemes .The elliptic curve schemes are able to use less storage and hence are recommended for applications which involve less battery power in adhoc and emergency situations.

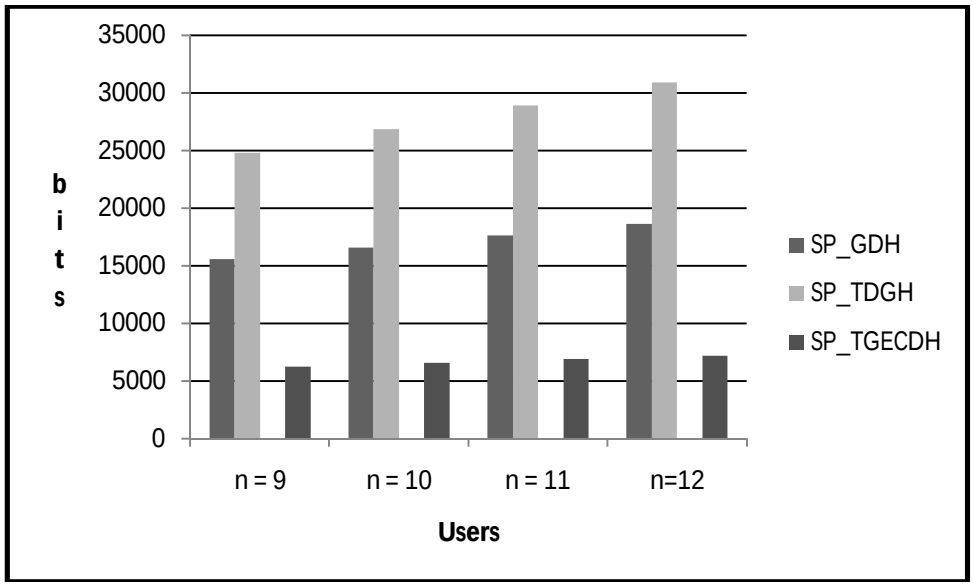

Figure 4 : Storage Cost Less Number of Users in Each Class

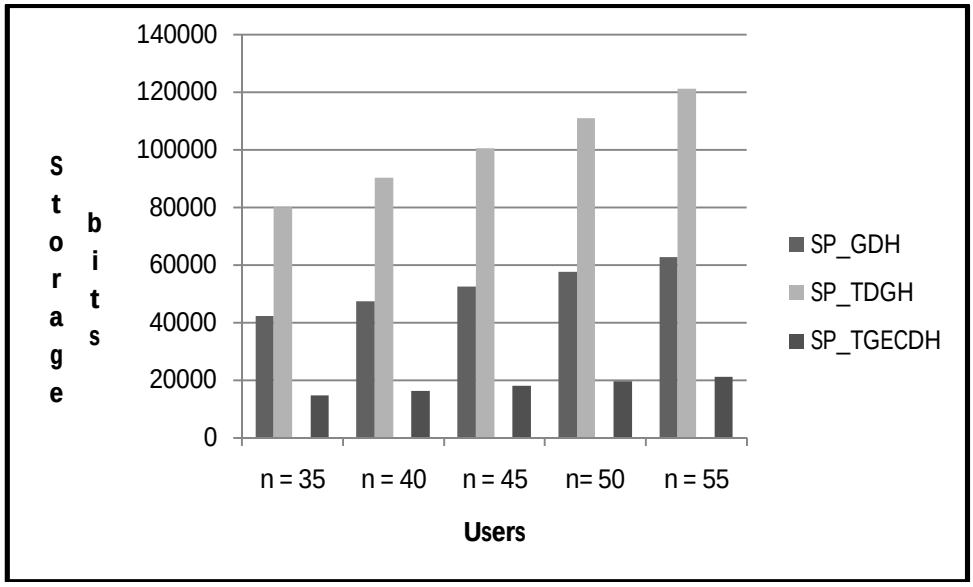

Figure 5 : Storage Cost for More Number of Users in Each Class

Communication Cost

Communication cost for the proposed scheme is the sum of communication cost for performing the communication semantics to enable Hierarchical Access Control and the communication cost for the respective contributory key agreement scheme . The communication semantics in the Upper Layer and Lower Layer are needed for communication of encrypted / decrypted messages within the sender Security Class and up to three messages for enabling the ancestor classes to receive the message. There is an additional cost for communication of encrypted / decrypted messages within each ancestor security class to make known the messages. However these messages are common for all the schemes and hence not considered for the comparative analysis. The communication cost for the key establishment in the CKA schemes is considered as the critical contributor for the communication cost and hence is taken as the basis for the comparative analysis.

Communication cost for the key establishment in the CKA schemes depends on Number of rounds, Number of messages and Size of a message. Communication costs needed for the group key agreement protocol in terms of number of messages are given in Table 7. Assuming, there are n ($n \geq 2$) members participating in the security class.

TABLE 7: Protocol Comparisons – Communication Analysis

Protocol	Event	Rounds	Total Message
SP-GDH	Join	n	n
	Leave	n-1	n-1
SP-TGDH	Join	2	3
	Leave	1	1
SP-TGECDH	Join	2	3
	Leave	1	1

Where

n … is the number of members in the group,

Let K_i and PK_u indicates the private key and the public key length and r be the overhead of each message. The key sizes used in the calculation are shown in Table 6. The following parameters and formulae as discussed by Yong Wang et.al [11] has been used in calculating the communication cost . The bandwidth is 11Mbps and the message overhead r =192 bits which is the length of a TCP header and each key tree node needs c=24 bits for storage when broadcasting. The frames error rate is p=8.70% Where c is the number of bits required to represent the key tree.

Let, TML_J denotes the total message length when n users establish a group key and TML_L denotes the total message length when the remaining n-1 users rebuild the group key after an existing member leaves.

In Group Diffie-Hellman (GDH / GECDH), the total message length for n users to generate the shared key can be calculated as follows:

$$TML_J = \frac{PK_u}{2} n^2 + \left(\frac{3PK_u}{2} + r\right) n - 3PK_u \quad (6)$$

When a member leaves the group, the remaining n-1 users need to rebuild the group key as n-1 users build the group key. Thus,

$$TML_L = \frac{PK_u}{2} (n-1)^2 + \left(\frac{3PK_u}{2} + r\right)(n-1) - 3PK_u \quad (7)$$

In tree based group Diffie-Hellman protocol (TGDH/TGECDH) , join and leave have different processing loads. When a new participant joins a group of size n, three messages are required.

- The new user broadcast its join request.
- The group controller node changes its contribution and broadcasts the key tree and the public key of the nodes to the joining member.
- The new member acts as a group controller node and broadcast the new public keys to remaining users.

The message size for a new user to join a group of size n is equal to:

Message size = $2hPK_u + (2n-1)c + 3r$

Where h is the height of the binary tree and thereby $h = \lceil \log_2 n \rceil$

Therefore, the total message length to build a group of n users to generate the group key can be calculated as:

$$TML_J = 2S_n PK_u + (n^2 - 1)c + 3(n-1)r \quad (8)$$

Where

$$S_n = (n+1)h - 2^h + 1 \text{ and } h = \lceil \log_2 n \rceil$$

When a member leaves the group in TGDH protocols, the group controller needs to generate a new private key, recalculate the agreed keys and public keys along the key path and broadcast the new public key. Thus, the message size for one member leave is equal to

$$TML_L = hPK_u + r$$

The communication time can be calculated as:

$$\text{Communication Time } t = \frac{TML}{B} \frac{1}{1-p} + \frac{sd}{3x10^8} \quad (9)$$

Where TML is Total message length for Join or leave the group. B indicates the bandwidth of the network, d the maximum distance between two participants, s the number of messages to build a group key for n parties and p the probability of frames in errors.

Compared with the transmission time, the propagation delay $\frac{sd}{3x10^8}$ is very small.

Thus, approximately the estimated communication time is

$$t = \frac{TML}{B}\frac{1}{1-p} \quad (10)$$

The communication cost is calculated for the various schemes and shown in Figure 6 and Figure 7. It is seen that SP-TGECDH scheme performs better compared to other schemes.

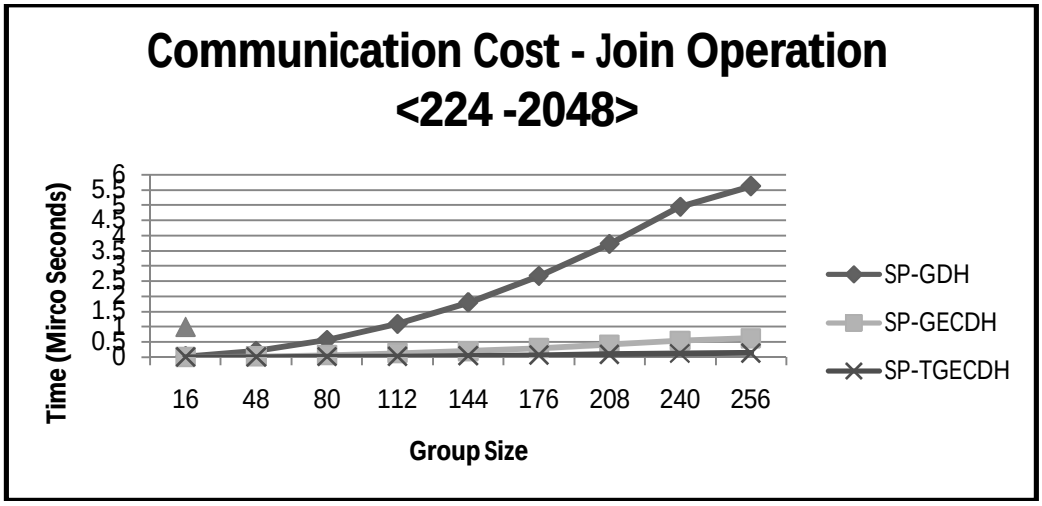

Figure 6 : Communication Cost – Join Operation

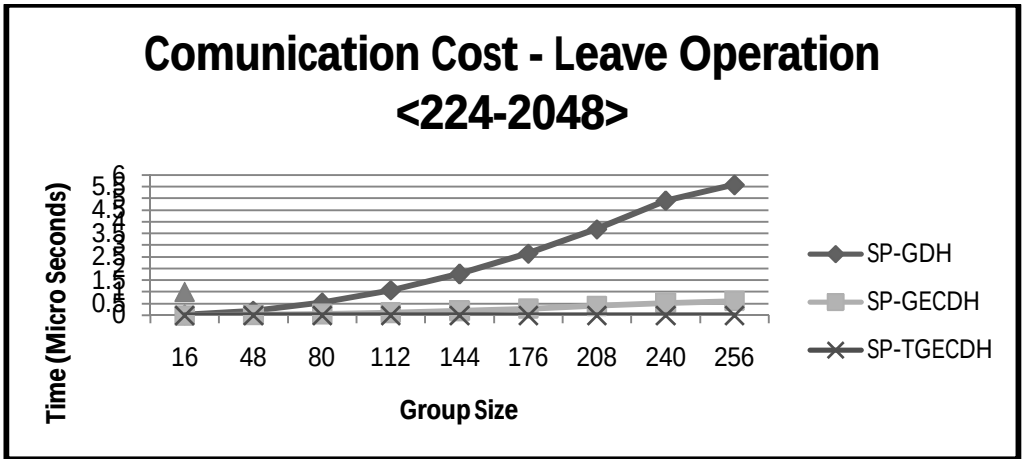

Figure 7 : Communication Cost for Leave Operation

Conclusion :

The proposed scheme is able to satisfy all the requirements i.e providing high dynamicity using user dynamics in level 2 and scalability through class dynamics in layer1. in addition it satisfies confidentiality through upward secrecy, downward secrecy. forward secrecy, backward secrecy and provides access control in the hierarchical group.

References

[1] S. G. Akl and P. D. Taylor,1983," A cryptographic solution to the problem of access control in a hierarchy", ACM Transactions on Computer Systems (TOCS), 1(3):239.248.

[2] X. Zou, B. Ramamurthy, and S. S. Magliveras, 2004, editors. Secure Group Communications Over Data Networks. Springer, New York, NY, USA, ISBN: 0-387-22970-1.

[3] R. S. Sandhu, 1988," Cryptographic Implementation of a Tree Hierarchy for Access Control", .Information Processing Letters, 27(2):95–98, 1988.

[4] M. J. Atallah, K. B. Frikken, and M. Blanton," Dynamic and efffcient key management for access hierarchies". ACM CCS'05, pages 190.202, Nov. 2005.

[5] C. Blundo, A. De Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung,1993," Perfectly-Secure key distribution for dynamic conferences", In Advances in Cryptology. Proc. of Crypto 92 (Lecture Notes in Computer Science,740), Springer-Verlag, pages 148- 168.

[6] M. L. Das, A. Saxena, V. P. Gulati, and D. B. Phatak,2005," Hierarchical key management Scheme using polynomial interpolation", SIGOPS Operating Systems Review, 39(1):40.

[7] X.Zou and L.Bai ,2008, " A New Class of Key Management Scheme for Access Control in Dynamic Hierarchies " , International Journal of Computers and Applications, Vol. 30, No.4, 331-337.

[8] W. Diffie and M. E. Hellman. , 1976, "New directions in cryptography",. IEEE Transactions on information theory, IT-22(6):644- 654.

[9] Kim.Y,Perrig.A and Tsudik.G , 2004, "Tree-based group key agreement", ACM Transactions on Information Systems security,7(1),pp. 60-96.

[10] R. Aparna, and B. B. Amberker,2009, " Analysis of Key Management Schemes for Secure Group Communication and Their Classification", Journal of Computing and Information Technology – CIT 17, 2009, 2, 203–214.

[11] Yong Wang et.al ,2006," The performance of Elliptic Curve Based Group Diffie-Hellman protocols for Secure Group Communication Over Ad Hoc Networks",IEEE ICC 2006 proceedings,pp.2243-2248.

[12] Kumar, K., Nafeesa Begum, J. and Dr. Sumathy, V.2011, "Cost Effective Region-Based Group Key Agreement Protocol for Ad Hoc Networks Using Elliptic Curve Cryptography", Lecture notes in Computer Science (LNCS), Vol.7135, pp.505-514, 2011. (ISSN: 0302-9743

[13] Hans Chalupsky, Yolanda Gil, Craig A. Knoblock, Kristina Lerman, Jean Oh, DavidV.Pynadath,Thomas A. Russ, And Milind Tambe ,2002,, "Electric Elves-Agent Technology For Supporting Human Organizations ", AI Magazine Volume 23 Number 2 .

[14] Arkadii Slinko,2006, "Symmetric Polynomials" , New Zealand Mathematical Olympiad Committee, http://www.mathsolympiad.org.nz

[15] Michael Z. Spivey1, Andrew M. Zimmer, 2008, " Symmetric Polynomials, Pascal Matrices, and Stirling Matrices, Linear Algebra and its Applications, Volume 428, Issue 4, Pages 1127–1134