International Journal of Pure and Applied Mathematics
Volume 117 No. 20 2017, 795-807
ISSN: 1311-8080 (printed version); ISSN: 1314-3395 (on-line version)
url: http://www.ijpam.eu
Special Issue

ijpam.eu

Dynamic and Collaborative Group Key Generation with Ternary Tree based Queue-Batch Algorithm

[1]V. Srinadh and [2]P.V. Nageswara Rao

[1]Computer Science & Engineering,

GMR Institute of Technology,

Jawaharlal Nehru Technological University,

Kakinada, Andhra Pradesh, India.

srinadh.v@gmrit.org

[2]Computer Science & Engineering,

GITAM Institute of Technology,

GITAM University,

Visakhapatnam, Andhra Pradesh, India.

nagesh@gitam.edu

Abstract

Now a days most of the communications are in the form of group which is called Group communication. In group communication the communication will be done with the help of one security key that is called Group Key. Even though it is secret key there are chances to crack the secret key by attackers. But if we consider rekeying i.e. the group key has to be changed as and when any member leave the group as well as any member join the group. This group communication can be represented by a hierarchical data structure called key tree. Rekeying mainly depends on the height of the tree. In hierarchical binary tree representation of group members height of the tree will be increased if the members are increased, therefore the rekeying generation requires much time. In compared to Binary tree if we use ternary tree the tree height will be less, so the rekeying operations require less time. This scenario is mainly having two advantages; firstly the communication will be more secure as the group key is going to change as and when a new members joining into the group and existing members leaving from the group; second one is the rekeying will be done in less time compare to binary representation of group members. In this paper we are going implement Queue-Batch Algorithm using ternary tree for group key generation.

Key Words: Group key, joining, leaving, Rekeying, security, Ternary tree, Queue-Batch Algorithm.

1. Introduction

In general Security is the state of being free from danger or threat or the state of being protected or safe from harm. Security is the degree of resistance to, or protection from, harm. Security is the Protection of a person, building, organization, or country against threats such as crime or attacks by foreign countries.

It is the practice of preventing unauthorized access, use, disclosure, modification, inspection, recording or destruction of information. Information is an asset to all individuals and businesses. Information Security refers to the protection of these assets in order to achieve C - I – A; where C stands for Confidentiality, I stands for Integrity and A stands for Availability.

Confidentiality: protecting information from being disclosed to unauthorized parties.

Examples:

- Personal: When submitted to a website, your personal data should only be used or accessed exclusively by designated staff in that company for the purposes agreed. No one else should be allowed to use your data for illegal purposes, or view the data out of curiosity.
- Business: Sensitive information, such as sales figures or client data, should only be accessed by authorized persons such as senior management and the sales team, and not other operations or departments.

Integrity: In information security, data integrity means maintaining and assuring the accuracy and completeness of data over its entire life-cycle. This means that data cannot be modified in an unauthorized or undetected manner. protecting information from being changed by unauthorized parties.

Examples:

- Personal: When submitted to a website, your personal data should not be altered in any way during data transmission, or by the website company.
- Business: Important documents or figures should not be changed or altered by unauthorized persons.

Availability: Availability of information to authorized parties only when requested.

Examples:

- Personal: You should be able to access and check your personal data kept on a website at any time.

- Business: Authorized senior management personnel should be able to access sales figures when needed; or clients should be able to access any of their data kept by the company when they request it.

Group Communication: The interaction of three or more interdependent members working to achieve a common goal is called Group communication. Group communication is a mode of communication in an organization, between employers and employees, and employees in teams/groups. Group communication can have effective results in case of marketing, where the communication is vital for selling and marketing products and product launches etc. Security plays vital role in any communication system especially in Group oriented communication. In group oriented communication system entire communication will take place with the help of one secret key which is called Group Key. This group key has to be modified whenever a new member joins into the group so that the joined members cannot access the previous communicated data. As well as group key has to be modified whenever an existing member leaves from the group so that the left members cannot access the further communication data. Changing the group key whenever a new member joins into the group and an existing member leaves from the group is known as rekeying. This group communication can be represented using key tree. If we use binary tree, height of the tree will be increased if the members are more, so that rekeying operation takes more time. Instead of Binary tree if we use ternary tree the tree height will be less, so that rekeying operation takes less time.

To achieve this we need a secure distributed group key agreement and authentication protocol so that people can establish and authenticate a common group key for secure and private communication. First key agreement protocol was proposed by Diffie-Hellman. It can guarantee the security of communication between the two users. Tree-based Group Diffie-Hellman (TGDH) is one of the protocols that extend the Diffie-Hellman protocol to a group key agreement protocol.

Diffie-Hellman protocol: Diffie-Hellman is an asymmetric key algorithm used for public key cryptography. The Diffie-Hellman algorithm was created to address the issue of secure encrypted keys from being attacked over the internet when in transmission, though using the Diffie-Hellman algorithm in distributing symmetric keys securely over the internet. He process works by two peers generating a private and a public key. Peer A would send it's public key to peer B and peer B would send it's public key to peer A. Peer A would then use the public key sent from peer B and it's own private key to generate a symmetric key using the Diffie-Hellman algorithm. Peer B would also take the same process as peer A and in turn produce the exact same symmetric key as peer A, though enabling them to communicate securely over the in-secure internet. Both peers can now encrypt, transmit and decrypt data using their symmetric keys.

2. Tree-based Group Diffie–Hellman Protocol

To efficiently maintain the group key in a dynamic peer group with more than two members, we use the tree-based group Diffie–Hellman (TGDH) protocol proposed in [9].Each member maintains a set of keys, which are arranged in a hierarchical binary tree. We assign a node ID v to every tree node. For a given node v, we associate a secret (or private)

Key Kv and a blinded (or public) key BKv. All arithmetic operations are performed in a cyclic group of prime order p with the generator α. Therefore, the blinded key of node can be generated by

$$BK_v = \alpha^{K_v} \bmod p \quad \text{..... (1)}$$

Each leaf node in the tree corresponds to the individual secret and blinded keys of a group member M_i . Every member holds all the secret keys along its key path starting from its associated leaf node up to the root node. Therefore, the secret key held by the root node is shared by all the members and is regarded as the group key[1].

The node ID of the root node is set to 0. Each non leaf node consists of either three child nodes whose node ID's are given by 3v+1,3v+2 and 3v+3 for its left child, middle child and right child respectively or two child nodes whose node ID's are given by 3v+1 and 3v+2 for its left child and middle child respectively . Based on the Diffie–Hellman protocol [3], the secret key of a non leaf node can be generated, in two cases, as follows

Case 1: Non Leaf Node with Two Child Nodes

Let v be the non leaf node whose two children are 3v+1 and 3v+2 then the secret key of v can be calculated by the secret key of one child node of v and blinded key of another child node of v [11]. Mathematically, we have

$$\left.\begin{aligned} K_v &= (BK_{3v+1})^{K_{3v+2}} \bmod P \\ &= (BK_{3v+2})^{K_{3v+1}} \bmod P \\ &= \alpha^{K_{3v+1}K_{3v+2}} \bmod P \end{aligned}\right\} \quad \text{..... (2)}$$

Case 2: Non Leaf Node with Three Child Nodes

Let v be the non leaf node whose three children are 3v+1, 3v+2 and 3v+3 then the secret key of v can be calculated by the secret key of one child node of v and blinded key of other two child nodes of v. For example with the help of secret key of node 3v+1 and blinded key of node 3v+2 we can generate secret key according to the equation (2); using this secret key and blinded key of node 3v+3 we can generate the secret key of the node v using the same equation(2). Applying equation (2) for two times can be replaced by equation (3). Out of

three child nodes we can use any child node's private key and public key of other two child nodes. All possible cases can be represented mathematically as follows:

$$K_v = (BK_{3v+3})^{((BK_{3v+2})^{K_{3v+1}} \bmod P)} \bmod P$$
OR
$$K_v = (BK_{3v+2})^{((BK_{3v+3})^{K_{3v+1}} \bmod P)} \bmod P$$
OR
$$K_v = (BK_{3v+1})^{((BK_{3v+3})^{K_{3v+2}} \bmod P)} \bmod P$$
OR
$$K_v = (BK_{3v+3})^{((BK_{3v+1})^{K_{3v+2}} \bmod P)} \bmod P$$
OR
$$K_v = (BK_{3v+2})^{((BK_{3v+1})^{K_{3v+3}} \bmod P)} \bmod P$$
OR
$$K_v = (BK_{3v+1})^{((BK_{3v+2})^{K_{3v+3}} \bmod P)} \bmod P \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

Fig.1 is a key tree, which is represented by using ternary tree[2], of a group with 12 members.

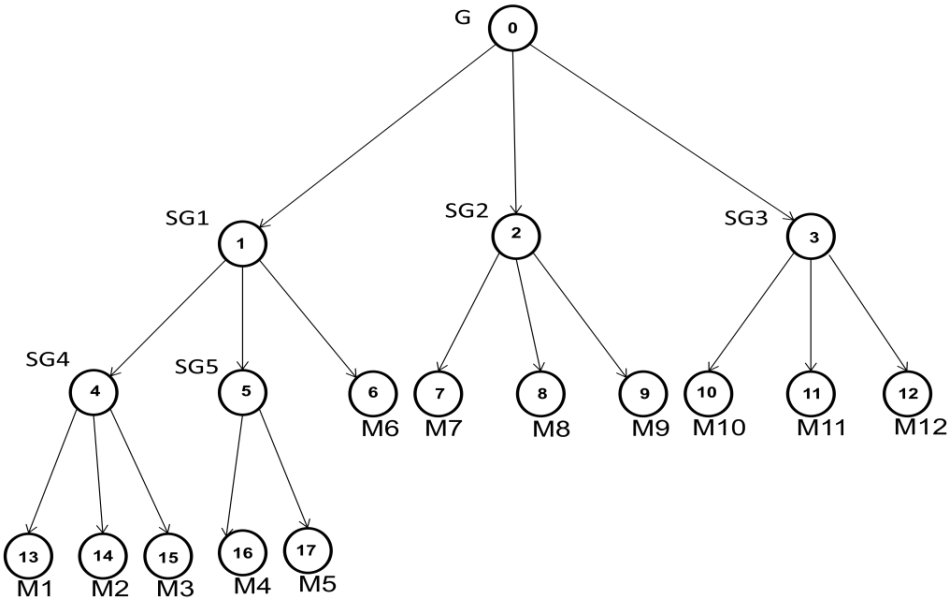

Figure 1: Key Tree for a Group with 12 Members

As the blinded keys are publicly known, every member can compute the keys along its key path to the root node based on its secret key by using the equation

(2) and (3). For example, M2 generates the secret key K_2 and it can request the blinded key BK_{13} from M1 and BK_{15} from M3. M2 generates the secret key K2 in two ways by using equation (3): either "first use the blinded key BK_{13} from M1 and generates new secret key and with this key and blinded key BK_{15} from M3, M2 can generate secret key K2" or "first use the blinded key BK_{15} from M3 and generates new secret key and with this key and blinded key BK_{13} from M21, M2 can generate secret key K2". This secret key K2 is considered as subgroup key SG4.If M1 is sponsor the same way SG4 can be computed and if M3 is sponsor the same way SG4 can be computed.

Now using SG4, SG5 and BK_6 the same way SG1 will be computed. Similarly SG2 and SG3 will be computed and finally from SG1, SG2 and SG3 the secret key K_0 i.e. G will be computed .This secret key K_0 or G is said to be group key[11]. Communication will be done using this group key until a new member joins into the group or existing member leaves the group.

3. Queue-Batch Algorithm

We have identified that earlier rekeying algorithms like ternary tree based rebuild algorithm and ternary tree based Batch algorithm perform all rekeying steps at the beginning of each rekeying interval. It leads to high processing overhead during update instance and so that the start of the group communication will be delayed. That is why we propose a more effective algorithm which we call Ternary tree based Queue-Batch algorithm. The idea in Ternary tree based Queue-Batch algorithm is to reduce rekeying load by preprocessing the joining members during the idle rekeying interval.

Ternary tree based Queue-Batch algorithm is divided into 2 phases:

(i) Queue-subtree.
(ii) Queue-merge.

Queue-subtree phase: Queue-subtree phase occurs whenever a new member joins the group during the rekeying interval. In this phase we append this newly joining member in a temporary key tree T′. Algorithm for Queue-subtree phase is as follows:

```
Algorithm Queue-subtree(T′)
{
if ( a new member is there to join)
{
 if (T′==NULL)          /* no new member in T′  */
Create a new tree T′ with this new member;
else       /* already there are some new members in T′ */
{
find the insertion node;
add the new node to T′;
elect the rightmost member under the subtree rooted at the siblings of the joining
node to be sponsor;
```

```
if (sponsor)
rekey the renewed nodes and broadcast new public keys;
}
}
```

For example consider a key tree as shown in the Fig(1). While generating group key if 2 members are joining into the group then subtree is created as shown in the fig(2).

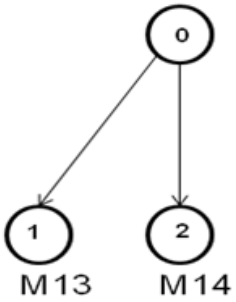

Figure 2: Subtree for Two Newly Joining Members

While generating group key if 4 members are joining into the group then sub tree is created as shown in the fig(3) :

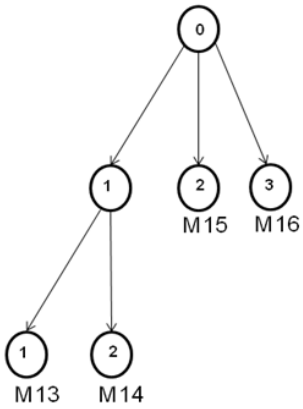

Figure 3: Sub Tree for Four Newly Joining Members

Queue-merge phase: Queue-merge phase occurs at the beginning of every rekeying interval. In this phase we merge the temporary tree T′ with the existing key tree T. Algorithm for Queue-merge phase is as follows:

Algorithm Queue-merge(T, T′ ,Ml, L**)**

```
{
if (L==0)  /* number of members want to leave is zero */
{
```

Add T′ to either the shallowest node of T such that the merge will not increase the height of resulting tree, or the root node of T if the merge to any locations will increase the height of the resulting tree.
}
else
{
Add T′ to the highest leave position of the key tree T;
Remove remaining L-1 leaving leaf nodes and promote their siblings;
}
Elect members to be sponsors if they are the rightmost members of the subtree rooted at the sibling nodes of the departed leaf nodes in T, or they are the rightmost members of T′ under the subtree rooted at the siblings of the joining node to be sponsor;
if (sponsor)
rekey the renewed nodes and broadcast new public keys;
}

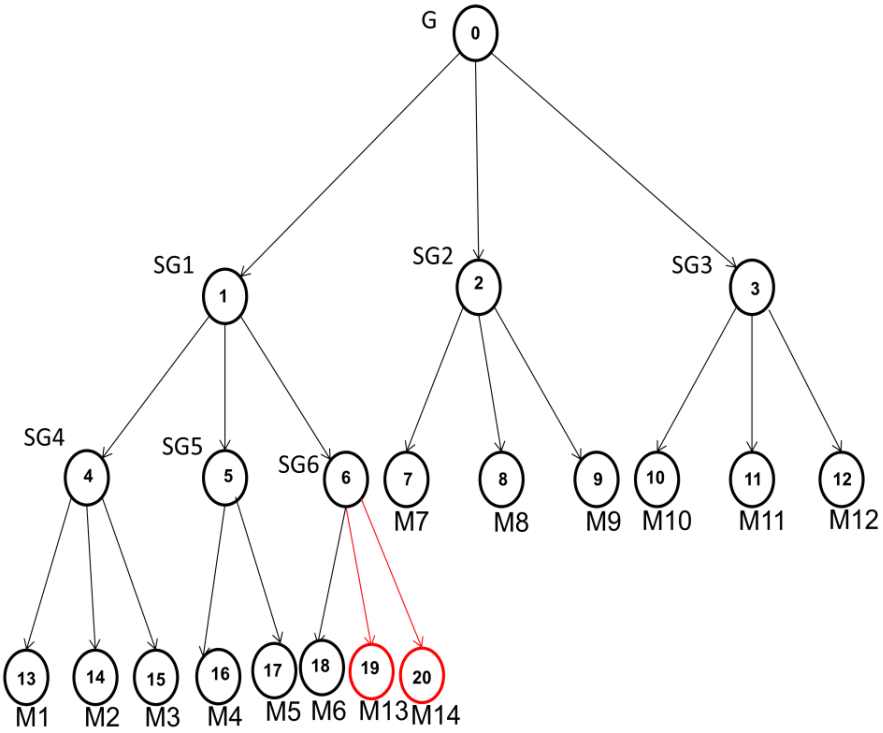

Figure 4: Key Tree After Merging of Existing Tree with Two Newly Joined Members

In the key tree for 12 members while generating group key if 2 members are joining into the group then the subtree is created as shown in the fig (2). Now after rekeying immediately next step is queue-merge phase. After merging of this newly created subtree which is shown in fig(2) with existing key tree which shown in fig(1) the resultant tree is as shown in the fig(4).

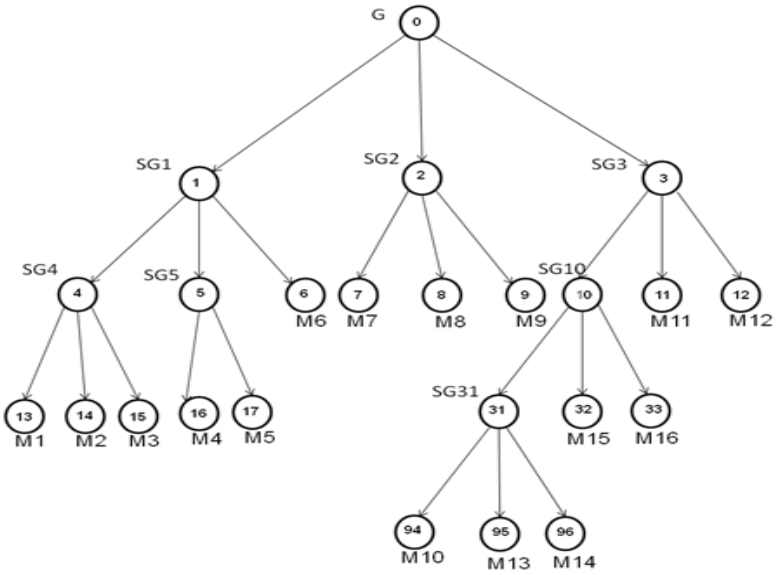

Figure 5: Key Tree After Merging of Existing Tree with Four Newly Joined Members

In the key tree for 12 members while generating group key if 4 members are joining into the group then the subtree is created as shown in the fig (3). Now after rekeying immediately next step is queue-merge phase. After merging of this newly created subtree which is shown in fig(3) with existing key tree which shown in fig(1) the resultant tree is as shown in the fig(5).

4. Rekeying

Rekeying (renewing the keys of the nodes) is performed for every single (multiple) join / leave event to ensure backward and forward confidentiality. A special member called sponsor is elected to be responsible for broadcasting updated blinded keys. Fig(6) represents the scenario that whenever one or more members are joining into the group the group key has to be rekeyed as well as one or more members are leaving from the group the group key has to be rekeyed.

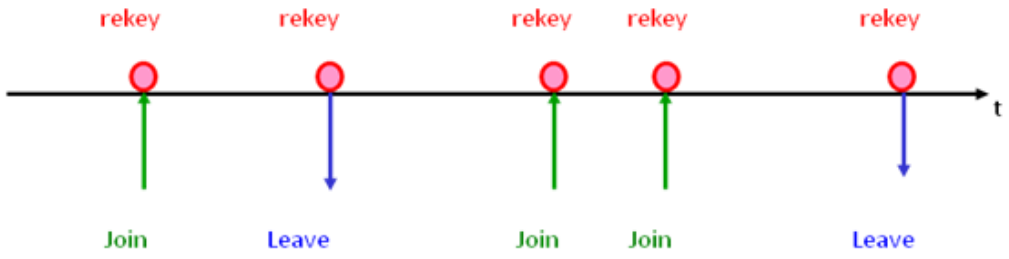

Figure 6: Key Tree After Merging of Existing Tree with Two Newly Joined Members

Consider the key tree of 12 members as shown in fig(1). If 3 members M3, M5

and M10 want to leave from the group and four members M13,M14,M15 and M16 want to join the group simultaneously then the resultant key tree is as shown in the fig(7).

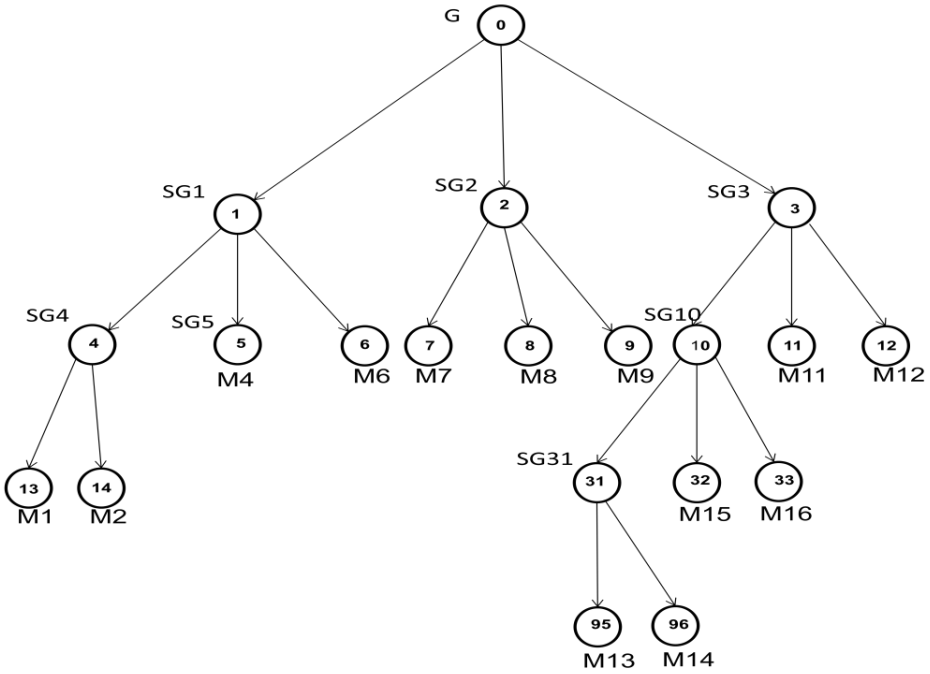

Figure 7: Key Tree After Leaving M3, M5, M7 and Joining M13, M14, M15 and M16

In this case the queue-subtree is created for four members M13, M14, M15 and M16 as shown in fig (3). As Node IDs of M3, M5 and M10 are 15,17and 10 respectively M10 is replaced by the queue subtree of joining four members; M3 and M5 are deleted directly but M4 is only child of node 5 so it has to promote to next higher level; thus the resultant keytree is as shown in the fig(7). Here SG2 will not change but remaining subgroup keys will change all will be calculated using equation (3) and finally the new group key will be computed and communication will be done using this new group key.

5. Mathematical Analysis and Experimental Results

The main idea of the Ternary tree based Queue-batch algorithm exploits the idle rekeying interval to pre-process some rekeying operations. When we compare its performance with Ternary tree based Batch algorithm, we need to consider only the rekeying operations occurring at the beginning of every rekeying interval. When there are no joining members, Ternary tree based Queue-batch algorithm is equivalent to pure leave case of Ternary tree based Batch algorithm. For J>0, the number of renewed nodes in Ternary tree based Queue-batch algorithm during Queue-merge phase is equivalent to that of Ternary tree

based Batch algorithm when J=1. Thus the expected number of renewed nodes is

$$E[R_{\text{Queue-batch}}] = \begin{cases} 0, & \text{if J=0, L=N} \\ \sum_{l=0}^{k-1} 3\hat{}l \left[1 - \frac{C(N - \frac{N}{3\hat{}l}, L)}{C(N, L)} \right] - L, & \text{if J=0, 0} \leq \text{L<N} \\ \sum_{l=0}^{k-1} 3\hat{}l \left[1 - \frac{C(N - \frac{N}{3\hat{}l}, L)}{C(N, L)} \right] - (L-1), & \text{otherwise} \end{cases}$$

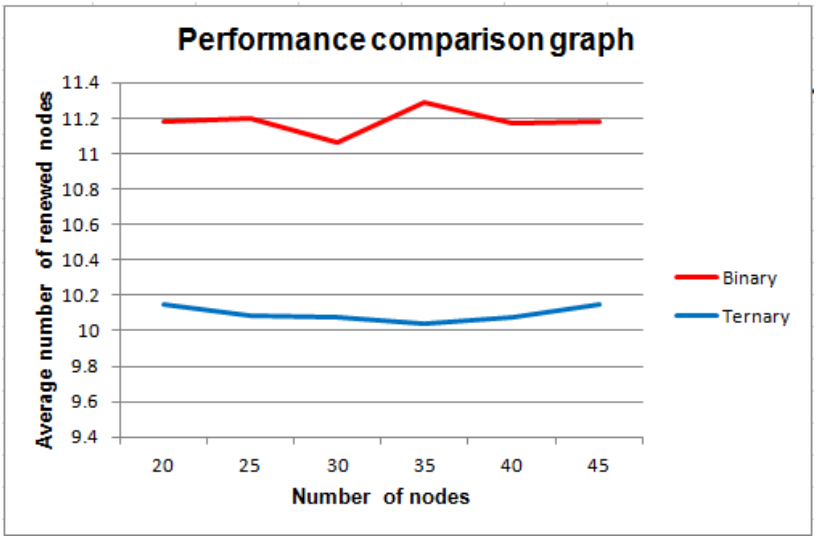

Figure 8: Mathematical Analysis: Average Numbers of Renewed Nodes at Different numbers of Joins

The graph in Fig (8) represents the performance comparison of rekeying of nodes using ternary tree based tree based Queue-Batch algorithm and binary tree based Queue-Batch algorithm.

6. Conclusion

In hierarchical binary tree representation of group members height of the tree will be increased if the members are increased, therefore the rekeying generation requires much time. In compared to Binary tree if we use ternary tree

the tree height will be less, so the rekeying operations require less time. This scenario is mainly having two advantages; firstly the communication will be more secure as the group key is going to change as and when a new members joining into the group and existing members leaving from the group; second one is the rekeying will be done in less time compare to binary representation of group members.

References

[1] Lee P.P., Lui J., Yau D.K., Distributed collaborative key agreement and authentication protocols for dynamic peer groups, IEEE/ACM Transactions on Networking (TON) 14(2) (2006), 263-276.

[2] Renuga Devi N., Mala C., Ternary Tree Based Group Key Agreement for Cognitive Radio MANETS, I.J. Computer Network and Information Security (2014), 10, 24-31.

[3] Diffie W., Hellman M., New directions in cryptography, IEEE Trans. Inf. Theory 22 (6) (1976), 644–654.

[4] Sherman T., McGrew D.A., Key establishment in large dynamic groups using one-way function trees, IEEE Trans. Software Eng., 29 (5) (2003), 444–458.

[5] Steiner M., Tsudik G., Waidner M., Key agreement in dynamic peer groups, IEEE Trans. Parallel Distrib. Syst., 11 (8) (2000), 769–780.

[6] Waldvogel M., Caronni G., Sun D., Weiler N., Plattner B., The versakey framework: versatile group key management, IEEE J. Sel. Areas Commun., 17 (9) (1999), 1614–1631.

[7] Wong K., Gouda M., Lam S.S., Secure group communications using key graphs, IEEE/ACM Trans. Netw., 8 (1) (2000), 16–30.

[8] Song B., Kim K., Two-pass authenticated key agreement protocol with key confirmation, In Proc. Indo Crypt., (2000), 237–249.

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

[10] Steiner M., Tsudik G., Waidner M., Key agreement in dynamic peer groups, IEEE Trans. Parallel Distrib. Syst., 11 (8) (2000), 769–780.

[11] Srinadh V., Nageswararao P.V., Ternary Tree Based Group Key Generation, IJETM International Journal of Engineering Technology and Management 2 (5) (2015), 1-04.