International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 6, No. 4, December, 2019, pp. 511-518
Journal homepage: http://iieta.org/journals/mmep

Decentralized Key Management Scheme Using Alternating Multilinear Forms for Cloud Data Sharing with Dynamic Multiprivileged Groups

Santhi Sri Kurra[1*], Veeranjaneyulu Naralasetty[2]

[1] Department of CSE, Vignan's Foundation for Science Technology & Research, Vadlamudi, Guntur-522213, Andhra Pradesh, India
[2] Department of IT, Vignan's Foundation for Science Technology & Research, Vadlamudi, Guntur-522213, Andhra Pradesh, India

Corresponding Author Email: drnvn_it@vignan.ac.in

https://doi.org/10.18280/mmep.060405

Received: 29 August 2018
Accepted: 25 October 2019

Keywords:
decentralized group key management-membership driven, alternating multilinear forms, CAMDH, cloud data, dynamic multiprivileged groups

ABSTRACT

Cloud Computing is a service oriented computing technology, is allows a group of people to work together and access data resources. Multi privileged Group key management has becoming a big challenging issue in the field of cloud data sharing. We have different group key management protocols which are distributed and centralized. But these protocols have drawbacks of single point of failure and bottleneck performance. Decentralized key management schemes proposed as trade-off between them. This paper proposes a Decentralized key management scheme using alternating multi linear forms for cloud data sharing with dynamic multi privileged groups. This method is to divide large group into many subgroups, each sub group has a group manager. Group manager manages the group, and keys are first distributed to the GM (Group Manger) and then GM can distribute to users in respective groups. The related session keys ought to be updates, if any cloud user needs to join/leave the group and change their privileges. But user joining/leaving the group as often as possible, users will change their entrance benefits called switching between various SGs. The proposed method needs just a single round of transaction for each leaving/Switching activity. This method also supports the dynamic formation and decomposition of Cloud service Groups. The analysis of proposed method is secure and has Reduces the Computational cost compared to existing scheme.

1. INTRODUCTION

Cloud data sharing service, which enables a group of members to cooperate to get to and change the common data, is a standout amongst the most mainstream and effective working styles in the undertakings. These days the group key administration is getting to be testing issue in the field of cloud data sharing. Conventional group key administration conventions are not adaptable for more groups. In this manner, some decentralized protocols are proposed. In any case, the cloud server isn't completely trusted, and its security could be imperiled by financial related reasons or caused by hacking and equipment mistakes. Along these lines, the security of convention relies upon the CMDH presumption.

To give content assurance in aggregate correspondence encryption of data resources is typical arrangement, secure group correspondence require giving in reverse security with the objective that new joining user unfit to unscramble past correspondence information successfully in the new joining group, and to give forward security so the pulled back user is never again prepared to decode the future correspondence data precisely in the left group. So, a great deal of keys should be rekeyed as regularly as conceivable in group key management. Group key organization is such a system to generate, distribute and update the keys safely and capably. Generally, Group key management protocols are classified into three which are

(1) Distribute
(2) Centralized
(3) Decentralized

1.1 Distribute group key management

Distributed group management scheme [1], allows group of people work together and they are responsible for new key generation and distributing them. It doesn't have any group controller, this makes the framework fault tolerant, users produce group key in a contributory way. This convention takes care of the issue of single point of failure and it decreases bottlenecks in the system in contrast with centralized schemes yet it has drawback to generating keys involves that every user know about the list of current users (Trust issues) and each member in the group does not have the computational resources to generate keys. Overall, this protocol is hard to manage the group.

1.2 Centralized group key management

In a centralized system [2], a single entity is responsible to complete group communication. Using this single entity we have to manage key generation, distribution. Approach, for example, Logical Key Hierarchy (LKH) decreases the overhead of rekeying. The huge challenges of a brought together plan incorporate Scalability overhead: As the achievement of gathering correspondence depends upon the single concentrated element, the errand of rekeying transforms into an overhead when the gathering size increments. Capacity overhead: The quantity of keys to be secure for a session. Single purpose of disappointment. Keeping up forward and in reverse security another client joins or an old client leaves the gathering. Agreement freedom: Co-undertaking among removed clients to coordinate and offer their own piece of information to recoup access to the gathering key.

1.3 Decentralized group key management

Decentralized group key management protocols [3], is to divide large group into many subgroups, each sub group has a group manager. Group manager mange the users joining/leaving operations. Keys are first distributed to the GM (Group Manger) and then GM can distribute to users in respective groups. The related session keys should be updated, if any cloud user wants to join/leave the group. It uses the advantages of both centralized and distributed schemes. This protocol reduces the load on KDC. This is accomplished by part the large group into a many subgroups and each subgroup is control by its own subgroup controller. This methodology solves the issue of a single point failure. This protocol is further categorized into two different protocols which are membership driven protocol and time driven protocol. Member ship protocol is when group user wants join /leave the group, the particular session key should be update. Whereas time driven protocol is carried out at certain time intervals.

The data is scrambled with same session enter in conventional single-privileged group correspondence. In the event that users' needs to join/leave the group then the specific session key ought to be refresh. Exactly when user have different access benefits with different data assets, multiprivileged assemble show up, which turns out to be new difficulties in aggregate key administration. From data proprietor's perspective, client who get to a particular data from data gathering (DG). In addition, from clients' perspective, client who get to practically identical data from advantage benefit gathering (SG). Be that as it may, client joining/leaving the gathering a great part of the time, client will change their passageway benefits called trading between different SGs. The security in multiprivileged total key organization is harder to ensure. As the data resources are encoded by SKs and clients are given specific SKs as exhibited by their advantages, assemble key organization ought not just guarantee that no client can get to the data past their benefits yet what's more be adequately versatile to oblige user's exercises of joining, leaving, and exchanging.

The rest of this paper is organized as follows. Section II describes the related work. Section III explains the details of preliminaries, Section IV describes the proposed method then, Section V explains the security our scheme, and section VI estimate the performance of proposed scheme. Finally, Section VII concludes the paper.

2. RELATED WORK

Group Key management with multiprivileged communication is becoming a very challenging issue in the field of cloud data sharing, because of secure dynamic membership. Nowadays user's wants access different data resources and they want to switch between two different groups, but traditional key management schemes cannot be extended multiprovileged groups. Here we describe some existing key management schemes, limitations of existing algorithms and explains proposed version.

Sun et al. [4] present a multi-group key administration protocol, this plan is likewise change key each time when participation change utilizing coordinated key chart (IKG) which incorporates the key trees into an incorporated key diagram (IKG). The presentation of IKG removes enter redundancies existing in the autonomous gathering key tree conventions and along these lines is helpful to reducing the rekeying overhead. In perspective of IKG, Wang et al. [5] propose an ID-based key administration contrive in to improve the rekeying capability. This convention uses an acceptance technique with the help of the key distinguishing proof (ID) to empower the client to enroll the keys without any other individual's information, making that the key dispersion focus (KDC) simply needs to multicast a couple of IDs and the keys that can't be figured by the clients. In any case, the introduction of IKG extends the degree of key sharing among clients, which annoys the issue of 'one impacts many'.

Wang et al. develop an encryption conspire for multiprivileged bunch correspondence [6] unites a convey encryption and a key-arrangement attribute based encryption system with a nonmonotone get the opportunity to structure, which can give gainful key repudiation. Furthermore, Angamuthu et al. [7] propose a character based convey encryption plot. In these perfect models, if the participation changes in a solitary SG, the relating information resources are re-mixed by the properties of the information assets and the present ID set of the clients, and after that the new unscrambling key for each client can be contemplated for the clients. As needs be, no rekeying messages are transmitted for keeping up forward/in reverse security. In any case, these plans rely upon a concentrated substance to control the whole gathering, which can be a danger of a singular motivation behind single purpose of disappointment and execution bottleneck. To unravel these issues, some key administration plans are proposed for the circulated conditions. There is no express KDC in conveyed key administration conventions. In addition, the keys can be made in a contributory way, where all people contribute their own offer to calculation of keys.

Sun et al. [8] were the first to use their proposed IKG to contributory condition. An organized gathering key understanding designs is also proposed in light of IKG for disseminated contributory condition in, which can bolster dynamic improvement and deterioration of SGs. The outsourcing information organization administrations [9] give the flexible, versatile and sensible answer for adventures and Internet clients to manage their information assets. The open encryption conventions [10, 11] have been proposed to anchor the insurance of the outsourced information. Regardless, these innovations can't safely share the keys in the checked clients. To engage the protection safeguarded cloud information sharing administrations, the current explores generally fixate on the gathering key administration to achieve the passageway control for the scrambled cloud information. The current inquiries about can be isolated into two arrangements: symmetric key based gathering key administration calculation and hilter kilter key based gathering key administration strategy.

A disseminated key diagram is planned for rekeying in multiprivileged gatherings [12], a disseminated key diagram is planned for rekeying in multiprivileged gatherings. In light of the key diagram, a scattered rekeying plan is delineated, which relies upon some passed on SG servers to manage the enrollment in singular SGs. In each joining/leaving/trading assignment, each SKs, ought to be refresh, will be counseled between the influenced SGs. Along these lines, there are a few transactions in one rekeying action in light of influenced SKs. At last, the organized SKs are passed on to the SG servers through secure channel. The Diffie-Hellman key trade plot [13] is the fundamental open key based key organization scheme. As a rule society key based crypto structure, open key affirmation is one of the major troubles. The PKI based open key crypto structure achieves open key validation through confining the confirmation with the overall population key. In this administration mode, the validations are issued and directed by the confided in outsider, anyway it also carries the huge cost of managing the confirmation summary to the CA. To address this issue, A. Shamir proposed the Identity Based Cryptosystems (IBC) [14], in which the trusted in Private Key Generator (PKG) utilizes the one of a client recognizable proof and the structure ace key to make the client's private key. In the IBE convention, the PKG gets every one of the clients of open and private keys so this model tackles issue of security in aggregate correspondence.

Many group key administration conventions were proposed for dealing with the dynamic participation and circulate keys to approved clients. Be that as it may, the greater part of the cloud clients are relied upon the cloud server for getting to the outsourced information, however cloud server is semi-trusted. The key organization frameworks [15, 16] rely upon Logical Hierarchical Graph (LHG) and Logical Hierarchical Tree (LHT). Which are the great courses of action supporting the dynamic access control approach. The certified client can reason all the supported keys just by understanding the unassuming number of keys. Regardless, these frameworks require the server to welcome the key finding, so the security still rely on the cloud server.

Conventional gathering key administration plans are not adaptable for substantial gatherings, along these lines, some decentralized plans are proposed. The administration of a substantial gathering is partitioned among subgroup servers, this scheme is to reduce the load on KDC, the central entity. This method is to divide large group into many subgroups, each sub group has a group manager. Keys are first distributed to the GM (Group Manger) and then GM can distribute to users in respective groups. The related session keys ought to be refreshed, if any cloud client needs to join/leave the gathering and change their benefits. But client joining/leaving the gathering often, clients will change their entrance benefits called exchanging between various SGs. The proposed technique needs just a single round of transaction for each leaving/Switching activity. This strategy additionally bolsters the dynamic arrangement and decay of Cloud benefit Groups.

3. PRELIMINARIES

In this section we describe the alternating multilinear forms and Computational Alternating Multilinear Diffi-Hellaman (CAMDH) assumptions

3.1 Alternating multilinear form

We say that a map $e_y: G_1^y \to G_2$ is an y-alternating multilinear map, if it satisfies following possessions

(1) G_1 and G_2 are finite cyclic groups with same prime order p;

(2) If $a_1, a_2 \dots, a_y \in Z_p$, and $A_1, A_2, \dots, A_m \in G_1$, then $e_y(A_1^{a_1},\dots,A_y^{a_y}) = e_y(A_1,\dots,A_y)^{a_1 \cdots a_y}$;

(3) The map e_y is non-degenerate in the following logic if $g \in G_1$ is a generator of G_1 then $e_y(g, \dots, g)$ is a generator of G_1

3.2 Computational alternating Multilinear Diffi-Hellman (CAMDH) assumption

The computational alternating multilinear Diffi-Hellman(CAMDH) problem says , for an m-alternating multilinear map $e_m: G_1^y \to G_2$, given $g, g^{a_1}, \dots, g^{a_y}$ in G_1 where $a_1, a_2 \dots, a_y \in Z_p$, compute $e_y(g,\dots,g)^{a_1 \cdots a_y}$ in G_2

This assumption means the CDMH problem hard. Then the proposed method security is based on this assumption.

4. PROPOSED METHOD

The proposed method is Decentralized key management scheme using alternating multilinear forms for cloud data sharing with dynamic multiprivileged groups. The scheme is to reduce the load on KDC. This method is to divide the large group into sub groups, here every group has one group manager he/she manages the group joining/leaving/switching operations. The session Keys are first distributed to the GM (Group Manger) and then GM can distribute the keys to users in respective group. Figure 1 shows the decentralized Group Key Management for cloud data storage architecture.

Once cloud user wants to join, leave and switch the group, the particular keys are updated, here multiprivivilegd group communication is allowed, when cloud users to wants to switch another service group. Cloud user can control two different groups. This method solves the problem of single point failure. In this Decentralized group key management we are using membership driven rekeying here every time the session keys are update, when cloud user wants join/leave/switch the group.

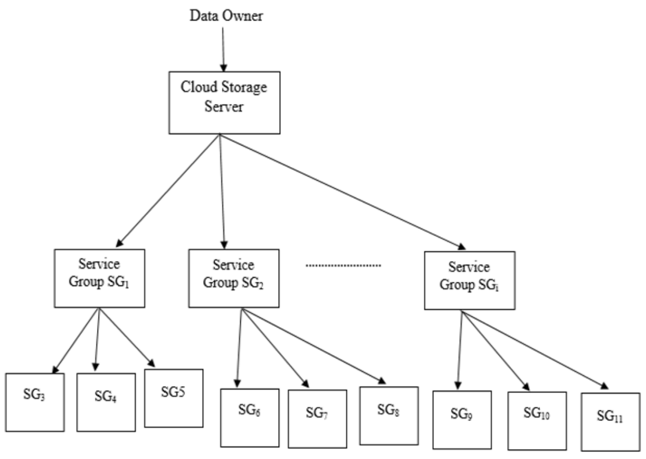

Figure 1. Decentralized group key management for cloud data sharing

The groups are indicated by SG_1, SG_2, SG_3,…, SG_n, …, where n is a prime number. We are utilizing Tensor forms to decrease rekeying cost. Give G_1 and G_2 a chance to be limited cyclic groups of a similar prime request$p$, and $g$be a generator of G_1 . An m -multilinear map $e_y: G_1^y \rightarrow G_2$ and a hash function $H: G_2 \rightarrow \{0,1\}^*$, these are decided for all SG managers, where r is sufficiently expansive. What's more, a random secret $s \in [1.p-1]$ is chosen. The group manager trough get system parameters(g, g^s), its public/private keys (Q_n, V_n) and general society keys of the other SG$_m$ chiefs $\{Q_m\}_{m \in D_R}$here, $V_n = Q_n^s$and $D_R$means the ID set of the SGs aside from SG$_{n.}$ To ascertain whether sessions keys are updated or not, the session key is distinguished by (A, B).

The proposed method consider a Cloud data sharing system consists of 4 entities which are

(1) Data owner
(2) Cloud Service Group
(3) Group Manger
(4) Cloud users

4.1 Data owner

A person has some data, he/she wants to store data in cloud server and share it with authorized users, and he /she become a data owner. The users who are shares this data, they are divided into several groups, cloud server is authorized this data to groups. Users can access outsourced data if they are in different groups.

4.2 Cloud storage server

Cloud server is to provide storage services and acts as proxy to generate and distribute keys to group managers. Cloud server first distribute the keys to group mangers and then group manager should distribute the keys to users in that respective group.

4.3 Group manger

Group Manger role is to take charge of user joining, User leaving, and user switching. Group manager updates keys every time if any user join/leave the group. The Group manager is the admin, he/she has the records of each and every process in the cloud that is when and which users upload and download data. Also the user belongs to which group etc.

4.4 Cloud users

The cloud users can access the cloud data when they are authorized to data owner because the data in cloud is encrypted so users should get key to access the data.

The proposed method is Decentralized Key Management Scheme using Multilinear Forms for Cloud Data Sharing with Dynamic Multiprivileged Groups allows following

(1) Users joining the group
(2) User leave the group
(3) Dynamic formation and decomposition of the group
(4) User switch the group

(1) User Joining

In this scheme we are using logical key hierarchy for key management, if cloud user wants to join the SG (Service Group), the group manager should update group key. Because the new user communicates with previous users to encrypt the data. The group manager distributes new key to cloud user through secure channel.

The new key GK_n' is generated as $GK_n' = f(GK_n)$ by the SG_n Manager

Here GK_n' is updated version of GK_n and f is a one-way function

At the same time, the group manger updates the session keys it carries as $SK_{(A,B)}' = f(SK_{(A,B)})$ where $SK_{(A,B)}'$ is updated version of $SK_{(A,B)}$. Eventually, the new session keys are encrypted with $GK_n'$and distribute to the users in SG_n. Also, the SG_n manager should inform the other SG managers to update the mutual SKs they hold. It demonstrates the signed message Sign (V$_n$, {join SG_n}). Here, Sign (k, M) denotes the signature of message M by the key k.

(2) User leaving process

When a cloud user wants to the leave the group SG_n, the SG_n manager generates a new GK_n' and then, group manager should distribute it to remaining users.It likewise advises the other SG manager by communicating Sign (V$_n$, {join SG_n}). In the wake of checking the signed message, at that point trough ought to check SKs update or not. To update the session keys safely and productively, a uniform rekeying material is consulted between the overcompensated service groups by utilizing alternating multilinear forms.

The SG managers who have the mutual session keys with SG_n, they can take an interest in the transaction of the rekeying material. Give x a chance to be the quantity of cloud clients associated with administration gatherings, which take an interest in the arrangement of a rekeying material, where $x < y$. Rather than a few exaggerated session keys there is just a single round for arrange a rekeying material. Each included SG administrator haphazardly chooses a mystery whole number $k_v \in [1, p-1]$ and indicates $Q_v^{k_v} \in G_1$ to the next influenced directors. Here, $v \in D_L$ where D_L the ID indicates set of the influenced SGs. At the point when an included SG director gets the messages, at that point it registers then R_v vas pursues

$$R_v = e_y(V_v^{k_v}, Q_{min}^{k_{min}}, \cdots, Q_a^{k_a}, Q_a^{k_b}, \cdots, Q_{max}^{k_{max}}, g^s, \cdots, g^s) = e_y(Q_{min}, \cdots, Q_a, Q_v Q_b, \cdots, Q_{max}, g, \cdots, g)^{k_{min} \ldots k_a, k_v k_b, X s^{y-x+1}}$$

Along these lines, all arbitrators acquire indistinguishable rekeying material from

$$R = H(R_{min}) = \cdots = H(R_{max}).$$

When knowing R, the influenced SG administrators can refresh the related SKs as

$$SK_{(A,B)}' = f(SK_{(A,B)} \oplus R)$$

Notice that SG_n director ought to be refresh all the session keys, on the grounds that the left client can knows these session keys, while the other included SG administrators just need to refresh the common SKs. At that point, each gathering administrator scrambles the new session keys with its gathering keys and disperses to aggregate clients.

(3) Dynamic formation and decomposition of SG

A cloud user needs to join the group to get to a few data assets, if there is no Service group having the benefit for joining the user then new Service group ought to be formed. The Session keys ought to be updated to related service group utilizing past SGs session keys. The previous key as

$SK'_{(A,B)} = f(SK_{(A,B)})$. The related session keys and other security parameters are unicast to the new service group through secure channel. The last advance is to multicast the session keys to cloud user in the new service group. The new service group can't know the past session keys, he/she knows SKs because of the irreversibility of one-way work in order to guarantee the backward security. In the event that there are no cloud users specifically service group, that group ought to be naturally decayed. The service group ought to be decomposed quickly if the last user takes off. The lazy-leave technique is utilized with a specific end goal to recreating service groups much of the time. This system is decomposed when there are no cloud user after an assigned period. The session keys are updated after disintegration of administration aggregate like rekeying activity of clearing out. The framework parameter for alternating multilinear forms ought to be updated first. A new secret $s \in [1.p-1]$ is selected randomly. The framework parameters (g, g^s), and its public/private keys (Q_n, V_n) are invigorated, and every SG trough gets the related parameters from key generator focus.

(4) Switching process

Multiprivivilegd group communication is allow cloud users to change access privileges according to their interests. Cloud user can control two different groups, if suppose user's switching process from SG_n to SG_m two different groups, user first leaves SG_n group and then join SG_m group. Here the important point is that the shared SKs between SG_n and SG_m need not be updated because the user access privileges have not been changed. Switch from SG_n means user should leave the group like leaving process, similarly switching into SG_m means user should joins the group like joining process.

Algorithm for Switching Process

```
Input: signed ($n$,$m$);optional {??}
Output: The updated keys;
/*A user switches from S$G_m$ $to$ S$G_m$
For all SG managers {
If verification signed ($n$,$m$)) passes Then
{
// S$G_m$ determines SKs, should be updated.
If ID(SG) = $n$ Then
{
For SKs which Group manager carry {
If $A$ mod $m$ !=0 then
$SK_{(A,B)}, flag = n$
//The SK is overdone by S$G_m$
}
}
```

SG$_m$ determines SKs, should be updated

```
//
If ID(SG)==m Then
{
For SKs which the SG manager carry
{
If A mod n !=0 Then $SK_{(A,B)}$. flag=$m$
// The SK is overdone by SG$_m$
}
}
```

The other SGs determine, SKs should be updated

```
If ID(SG) != n && ID(SG) != m Then
{
For SKs which the SG manager carry
{
If A mod (n*m) !=0 && A mod n=0 Then
$SK_{(A,B)}$. flag = n
If x mod (n*m) !=0 && A mod m=0 Then$SK_{(A,B)}.flag = m$
}
}
}
Else
The SG manger drops the signed message
}
```

The managers have the SKs overdone by SG$_n$ are associated with negotiation of rekeying material.

```
For the associated with the SG managers in negotiation
{
Show $Q_v^{k_v}$
//When receiving the other SGs' show messages
R$_v$ = $e_y(Q_{min}, \dots, Q_{max}, g, \dots, g)^{Q_{min},\dots,Q_{max} X s^{x-y+1}}$
R = H(R$_v$)
}
// The rekeying material R is Obtaining
For the overdone SG managers
{
For SKs which the SG manager carry
{
If $SK_{(A,B)}.flag == n$
$SK'_{(A,B)}$=$f(SK_{(A,B)} \oplus R)$
If $SK_{(A,B)}.flag == m$ Then
$SK'_{(A,B)}$=$f(SK_{(A,B)})$
}
}
```

5. SECURITY ANALYSIS

We give an analysis on the security in the regards of data confidentially and backward/forward security.

Data confidentially: In one rekeying process, session keys are gotten from past session keys and a discretionary rekeying material when a client leaves/changes from a SG utilizing a confined limit with respect to SG directors. Inside every SG, the SKs are multicast to the clients after scrambled with gathering key. Expect that enemies can tune in all surge hour gridlock. They can't hold any keys or the rekeying material since they have a place with no social events. Thusly, they can't procure SKs from the figure writings without GK. Along these lines, they can't acquire SKs from the cipher texts without GK. In addition, in light of the fact that the enemies are unaware of past SKs and the rekeying material, they can't conclude SKs without anyone else. In this manner, the data confidentiality is guaranteed.

Backward security: At the point when another cloud user joins or switches into a service group, new group key and new SKs will be concluded from the past keys by utilizing a one-way function as $GK'_n = f(GK_n)$ and $SK'_{(A,B)} = f(SK_{(A,B)})$ in light of the fact that the restricted limit is computationally irreversible in polynomial time, it is infeasible for the starting late joining cloud client to enlist the past keys from the new keys he holds. Besides, if another administration amass is surrounded, the new SG server moreover can't know the past SKs from the current SKs. In this way, in turn around security is kept up.

Forward security: Right when a client leaves or transforms from a SG, the SG servers will make another GK and transmit it to each one of whatever is left of the clients through a protected channel. Thusly, the left client can't obtain the new GK. What's more, the new SKs will be found from the past keys by using the limited limit and the agreed rekeying material as$SK'_{(A,B)} = f(SK_{(A,B)} \oplus R)$; in favour of the included SG servers. In this manner, they are communicated to the rest of the users in the wake of being encoded with new GK. In spite of the way that the noxious client holds the past SKs and the restricted capacity, he is 'in the not too distant past unfit to get new SKs without the rekeying material R. Since any private key is V_n and the relating discretionarily picked mystery k_n of the SG servers are dim to the harmful cloud customer, it is stunning for him to get R in light of the bother of CAMDH issue. Also, if a SG is disintegrated, the SG server can't organize another rekeying material to know the new sort of SKs in light of the manner in which that it can't pick up the new parameters in the trade. From now on, forward security is kept up.

6. EXPERIMENTAL RESULTS

The rekeying overhead is the rule concerning metric to assess the execution of key organization traditions. Inside every SG, SKs are directed as in ingle-favored gathering correspondence. Consequently, we center around inquiring about the rekeying overhead in the DG part in the midst of various rekeying outlines, autonomously. The correspondence costs diminish similar to the total size of transmitted messages. We compare our proposed scheme with the traditional group key management protocols. The computation costs of proposed method are analysed into different operations which are cloud user joining/leaving/switching operations.

Figure 2 explains the computation cost of key for a proposed and existing mechanisms. due to the structure of proposed mechanism give better computation cost than existing mechanism.

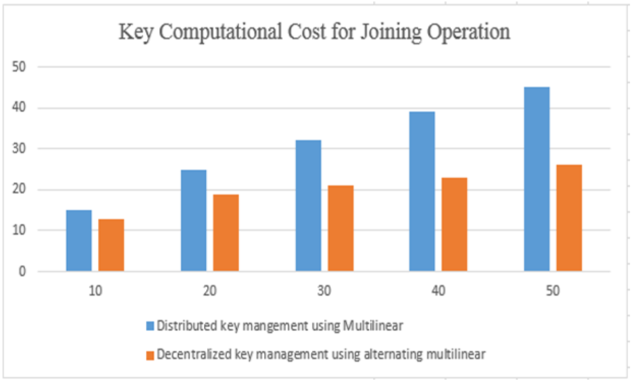

Figure 2. Key computational cost

Figure 3 explains the computation cost of key for a user to leve from group. proposed and existing mechanisms. due to the structure of proposed mechanism give better computation cost than existing mechanism.

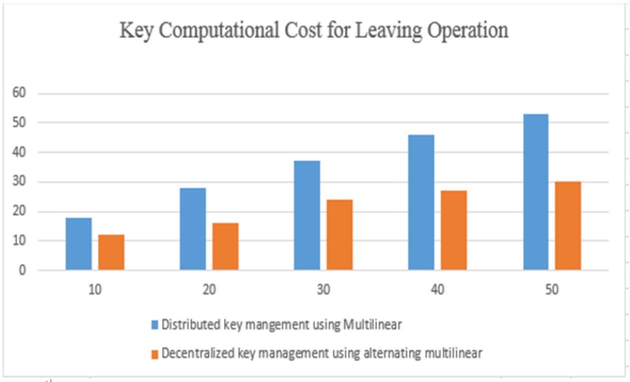

Figure 3. Key computational cost

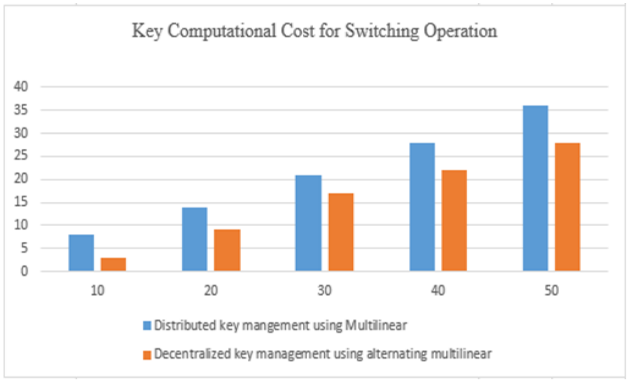

Figure 4. Key computational cost

Figure 4 describes the switching of auser from one group to another group. proposed and existing mechanisms. due to the structure of proposed mechanism give better computation cost than existing mechanism.

The service group managers/servers as a rule have high ability of computation, it is no issue for them to be in responsible for computation of session keys. In our proposed scheme there is no keys are distributing for rekeying process. Instead of many keys distributing, there is only rekeying material is distributed in leaving/leaving/switching operations. So that our proposed scheme reduces the communication cost efficiently. We compare proposed method with traditional key management method is distributed key management scheme through simulations using Java. The traditional group key management protocol takes more rounds negotiations because each session key is to be negotiated for one round, but proposed scheme takes only one round negotiation. So, the proposed scheme reduces the communication costs greatly. And also, the proposed scheme has great versatility when the group size turns out to be extensive.

7. CONCLUSION

This paper presents Decentralized key management scheme using alternating multilinear forms for cloud data sharing with dynamic multiprivileged groups. Which does not rely on Centralized and Distribute group protocols. In our method large group is divide into many subgroups, each sub group has one group manager. The group manager manages the user joining/leaving/switching operations. The related session keys ought to be refreshed, if any cloud user needs to join/leave the group and change their privileges. But user joining/leaving the group every now and again, user will change their entrance benefits called exchanging between various SGs. The proposed strategy needs just a single round of arrangement for each leaving/Switching task. This strategy additionally bolsters the dynamic development and deterioration of Cloud benefit Groups. This method overcome the drawbacks of single point of failure and bottleneck performance of centralized and distribute key management protocols. The analysis of proposed method is secure and decreases the computational time compare to existing algorithms.

REFERENCES

[1] Zhou, W., Yang, X., Wang, G.J. (2013). Distributed group key management using multilinear forms for multi-privileged group communications. Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on. IEEE. https://doi.org/10.1109/TrustCom.2013.78

[2] Seetha, R., Saravanan, R. (2015). A survey on group key management schemes. Cybernetics and Information Technologies, 15(3): 3-25. https://doi.org/10.1515/cait-2015-0038

[3] Zhou, W., Yang, X., Wang, G.J. (2016). Decentralized group key management for hierarchical access control using multilinear forms. Concurrency and Computation: Practice and Experience, 28(3): 631-645. https://doi.org/10.1002/cpe.3328

[4] Sun, Y., Liu, K.J.R. (2004). Scalable hierarchical access control in secure group communications. Proceedings of IN-FOCOM2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, 2: 1296-1306. https://doi.org/10.1109/INFCOM.2004.1357015

[5] Wang, G., Ouyang, J., Chen, H., Guo, M. (2007). Efficient group key management for multi-privileged groups. Computer Communications, Elsevier, 30(11-12): 2497-2509. https://doi.org/10.1016/j.comcom.2007.04.019

[6] Wang, G., Du, Q.S., Zhou, W., Liu, Q. (2013). A scalable encryption scheme for multi-privileged group communications. The Journal of Supercomputing (Springer), 64(3): 1075-1091. https://doi.org/10.1109/EUC.2010.96

[7] Muthulakshmi, A., Anitha, R., Rohini, S., Princy, K. (2012). Identity based privacy preserving dynamic broadcast encryption for multi-privileged groups. Recent Trends in Computer Networks and Distributed Systems Security Communications in Computer and Information Science, 335: 272-282. https://doi.org/10.1007/978-3-642-34135-9_28

[8] Sun, Y., Liu, K.J.R. (2004). Scalable hierarchical access control in secure group communications. Proceedings of IN-FOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, 2: 1296-1306. https://doi.org/10.1109/INFCOM.2004.1357015

[9] Song, W., Peng, Z., Wang, Q., Cheng, F., Wu, X., Cui, Y. (2014). Efficient privacy-preserved data query over ciphertext in cloud computing. Security and Communication Networks, 7: 1049- 1065. https://doi.org/10.1002/sec.824

[10] Xia, Z., Wang, X., Sun, X., Wang, Q. (2015). A secure and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans. on Parallel and Distributed Systems, 27(2): 340-352. https://doi.org/10.1109/TPDS.2015.2401003

[11] Fu, Z., Ren, K., Shu, J., Sun, J., Huang, F. (2015). Enabling personalized search over encrypted outsourced data with efficiency improvement. IEEE Trans - actions on Parallel and Distributed Systems, 27(9): 2546-2559. https://doi.org/10.1109/TPDS.2015.2506573

[12] Li, R., Li, J., Kameda, H. (2005). Distributed hierarchical access control for secure group communications. Proceedings of the 2005 International Conference on Communication, Networking and Mobile Computing (ICCNMC 2005), LNCS 3619, Zhangjiajie, China, pp. 539-548. https://doi.org/10.1007/11534310_58

[13] Liu, X.F., Zhang, Y.Q., Wang, B.Y., Yan, J.B. (2013). Mona: Secure multi-owner data sharing for dynamic groups in the cloud. IEEE Transactions on Parallel and Distributed Systems, 24(6). https://doi.org/10.1109/TPDS.2012.331

[14] Lu, R., Lin, X., Liang, X., Shen, X. (2010). Secure provenance: the essential of bread and butter of /data forensics in cloud computing. Proc. ACM Symp. Information, Computer and Comm. Security, 282-292. https://doi.org/10.1145/1755688.1755723

[15] Damiani, E., Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S., Samarati, P. (2005). Key management for multi-user encrypted databases. ACM Work - Shop on Storage Security & Survivability, pp. 74-83. https://doi.org/10.1145/1103780.1103792

[16] Vimercati, S., Foresti, S., Jajodia, S., Pataboschi, S., Samarati, P. (2007). Over-encryption: Management of access control evolution on outsourced data. Proc. of the 33rd International Conference on Very Large Data Bases (VLDB), pp. 123-134.

[17] Song, W., Zou, H., Liu, H.W., Chen, J. (2016.) A practical group key management algorithm for cloud data sharing with dynamic group. China Communications, 13(6): 205-216. https://doi.org/10.1109/CC.2016.7513215

[18] Rafaeli, S., Hutchison, D. (2003). A survey of key management for secure group communication. ACM Computing Survey, 35(3): 309-329.

[19] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R.H., Konwinski, A., Lee, G., Patterson, D.A., Rabkin, A., Stoica, I., Zaharia, M. (2010). A view of cloud computing. Comm. ACM, 53(4): 50-58. https://doi.org/10.1145/1721654.1721672

[20] Ateniese, G., Fu, K., Green, M., Hohenberger, S. (2006). Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Transaction on Information and System Security, 9(1): 1-30.

[21] Gu, X., Zhao, Y., Yang, J. (2012). Reducing rekeying time using an integrated group key. Journal of Communications and Networks, 14(4): 418-425. https://doi.org/10.1109/JCN.2012.6292248

[22] Sun, Y., Ma, H., Zheng, G., Yi, X., Pan, H. (2013). Multiple group shared key management for satellite multicast. Journal of Astronautics, 34(6): 824-832.

[23] Xia, Z., Wang, X., Sun, X., Wang, Q. (2015). A secure

and dynamic multi-keyword ranked search scheme over encrypted cloud data. IEEE Trans. on Parallel and Distributed Systems, 27(2): 340-352. https://doi.org/10.1109/TPDS.2015.2401003

[24] Yu, S., Ren, K., Lou, W. (2004). FDAC: Towards finegrained distributed data access control in wireless sensor networks. IEEE Trans. on Paralleland Distributed Systems, 22(4): 673-686. https://doi.org/10.1109/TPDS.2010.130