Analysis of Algorithms for Overlapping Resource Access Members in Cloud Computing

Amar Buchade[1], Rajesh Ingle[2]
Department of Computer Engineering, College of Engineering, Pune[1]
Wellesely Rd, Shivajinagar, Pune, Maharashtra 411005, India
Department of Computer Engineering & Pune Institute of Computer Technology[2]
Sr. No 27, Pune-Satara Road, Dhankawadi, Pune, Maharashtra 411043, India
(Email: arb.comp@coep.ac.in, ingle@ieee.org)
(Received Aug. 12, 2015; revised and accepted Nov. 12 & Dec. 15, 2015)

Abstract

In Cloud computing environment, resources include virtual machine, CPU and Storage. These resources are accessed by tenants. Group key may be used to access the resources securely. Group key is constructed using tree by considering tenants in a group. In existing scenario, different key trees are formed even if tenants are common among multiple groups to access the resources. This paper addresses the issues of overlapping tenants that accesses resources. If there are overlapping members in multiple groups, combined key trees may be formed. Through the analysis, it is observed that computational overhead is decreased by 24% if we combine the key trees than the separate key trees. It is also observed that key establishment time for combined key trees is less compared to separate key trees.

Keywords: Computational cost, key tree, resource, resource access membership matrix

1 Introduction

In cloud computing environment, resources are considered as virtual machine, CPU, storage. These resources are accessed by multiple tenants. Users of facebook may share data (multiple files) in multiple groups. Members in a group accesses the resources. To protect the resource from unauthorized users, each member in the group shares the partial information for forming the group key. In present scenario, group key is formed by considering separate key trees even if members are common to access multiple resources. It incurs redundant operations and thus leads to increase in computational cost and key establishment time. It causes delay in accessing the actual resource which an obviously violates the feature of cloud computing such as on demand resource access. Thus our paper proposes combined key trees formation and its analysis for the tenants overlapped in multiple resources.

Other example, member can be a part of multiple projects. Multiple tenants can be involved in multiple projects. For security purpose, members in a group form the group key to access the resource.

The other examples can be users of whatsapp/facebook sharing multiple files in groups. Many members can be overlapped in groups to access the files.

The solution is to combine key trees for resources which containing common members. We prove that our approach is efficient than the forming separate key trees for overlapping resources access members.

It reduces computational overhead and group key establishment time. It helps to support on demand resource access property of the cloud computing.

To form the group key, TGDH protocol is used [9, 12, 13]. More specifically our contributions are

1) Illustration of the algorithms through the examples.
2) Computation cost analysis of resource key formation for separate key trees and combined key tree in terms of total number of sequential exponentiation operations.
3) Formulation of key establishment time and analysis.

The paper is organized as follows. In Section 2, we describe about resource. Section 3 describes combining key trees algorithm in brief, Section 4 presents computational cost details in terms of sequential exponential and key establishment time, Section 5 covers results and analysis and Section 6 presents conclusion.

Assumption: Tree based Group Diffie Hellman Protocol [13] is used. Member who is acting as sponsor assumed to be trustworthy. The words "tenant" and "member" used alternatively.

2 Resource

2.1 Initializations

Let Resource group R = $\{R_1, R_2, R_3, R_4, ...R_n\}$;

Consider two resources R1 and R2.

Let $\{m_1, m_2, m_3, m_4, ...m_n\}$ be the members accessing resource R1.

Let $\{n_1, n_2, n_3, n_4, ...n_n\}$ be the members accessing resource R2.

It is possible to have members overlapped to access the resources R1 and R2.
Assume $R1 \cap R2 =$ cm where cm is number of overlapping members which accesses the resources R1 and R2.

2.2 Resource Key Tree

Figure 1 shows resource key tree with leaf nodes represents group members m1, m2, m3, m4 etc. [8].

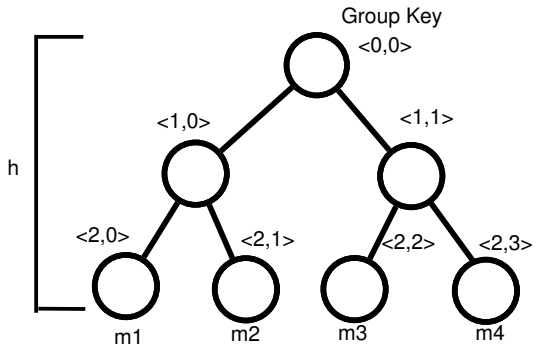

Figure 1: Resource key tree

Diffie and Hellman presented two party key exchange protocol called TGDH in 1976 [13].

- In TGDH [1, 6, 22, 23] group key is formed from bottom-up fashion.
- Members m1, m2, m3 and m4 have $\alpha 1, \alpha 2, \alpha 3$ and $\alpha 4$ private keys, respectively.
- Each member forms the public key called as blinded key.
- In this case, g is generator, p is prime number Member blinded keys are $g^{\alpha 1} \bmod p$, $g^{\alpha 2} \bmod p$, $g^{\alpha 3} \bmod p$, and $g^{\alpha 4} \bmod p$.
- Each member knows all keys on key path and all blinded keys. Key path of m2 includes the node at $< 1, 0 >$ and node at $< 0, 0 >$.
- Thus resource group key is formed as below.

$$g^{\alpha 1.\alpha 2.\alpha 3.\alpha 4} \bmod p$$

Let h is the height of the key tree. From Figure 1, h=2.

Number of exponential operations performed serially by the member are called sequential exponentiation operations. It is observed that number of sequential exponentiation required to form the group key = 2h - 2.

Thus to calculate group key at node $< 0, 0 >$, member m2 two sequential exponential operations mainly at node $< 1, 0 >$ and node $< 0, 0 >$ are required.

2.3 Resource Access Membership Matrix

Every tenant that is part of the resource, entry is made in its resource access membership [RAM] matrix also for any member that joins/leaves the resource. RAM matrix contains the following entries.

Members m_1, m_2, m_3, ..., m_n represents the entries in row wise.

Resources R_1, R_2, R_3, ..., R_n represents the entries in column wise.

$$\begin{bmatrix} 1 & 1 & \cdots & \cdots \\ 1 & 0 & \cdots & \cdots \\ 1 & 1 & \cdots & \cdots \\ 1 & \cdots & \cdots & \cdots \end{bmatrix}$$

It shows that there are $R_1, R_2, \ldots, R_n$ resources. Member m_1 accesses resource R_1 and R_2 i.e. overlapped to access the resources R_1 and R_2 while member m_2 is part of resource R_1, m_3 is a part of resources R_1 and R_2. Three dots (...) indicates entry either 0 or 1.

3 Combining Resource Key Trees Algorithm

In existing key management algorithm [11, 16, 20, 24], separate key tree is built for each resource, even if same members are accessing multiple resources. Buchade and Ingle gives combining key tree algorithm [4]. This algorithm takes consideration of resource access membership matrix. Key tree of overlapping members are formed as well as key tree of non overlapping members are formed. These key trees are combined. The combined key tree of overlapping members is rooted at the root node to reduce the height of the tree.

4 Computational Cost

This section covers the proof of sequential exponentiation and key establishment time for combined and separate key trees.

Lemma 1. *Total number of sequential exponentiation (SE) for separate key trees (SKT) required more than combined key tree (CKT).*

Proof. Total number of sequential exponentiation operations with SKT

$$[SESKT] = \sum_{i=1}^{NRK} (2h_i - 2)N_i \tag{1}$$

where $h_i = log(N_i)$ (property of binary tree); NRK = Total number of resource key tree; h_i = height of i^th resource key tree; N_i = Number of members of i^th resource key tree. Average height of each member, $h = \log(N)$.

Total number of SE due to overlapping members:

$$[SEOM] = \sum_{j=1}^{NOT} (2h_j - 2)O_j, \quad (2)$$

where $h_j = log(O_j)$, $h > 1$ otherwise SE=1; NOT = Number of trees formed due to overlapping members; h_j = height of j^th resource key tree; O = Number of overlapping members; Average height of each member, $h = \log(O)$.

SE due to combined key trees [SECKT]
= SE due to SKT - SE due to Overlapping members.

Thus it is observed that SE for separate key trees is more compared to combined key trees.

□

Lemma 2. *Key establishment time for SKTs is more than key establishment time for CKT. It depends on Number of members overlapped to access the resources.*

Proof. For SKT, total Number of members,

$$N = \sum_{i=1}^{NRK} N_i$$

Average height of each member, $h = \log(N/NRK)$; Time required to form the group key =

$$(2h - 2) * DH_t = (2log(N/NRK) - 2) * DH_t, \quad (3)$$

where, NRK = Number of resource key trees; N_i = Number of members of i^{th} resource key tree; DH_t = Time required to perform one Diffie Hellman Exponentiation operation.

For CKT, total Number of members = $N - cm$, where cm = Number of overlapping members; Average height of each member, $h = \log((N - cm)/NRK)$; Time required to form the group key =

$$(2h - 2) * DH_t = (2log((N - cm)/NRK) - 2) * DH_t \quad (4)$$

From Equations 3 and 4, it is observed that key establishment time for SKTs can be more than key establishment time for CKT. It depends on number of overlapping members. □

4.1 Single Join

Buchade and Ingle stated algorithm when tenant wants to access the resource [4]. If the joining tenant is not overlapped, it is added in the key tree as per TGDH. If it is overlapped to other resources forms the key graph. The details of the algorithm is given in detail through the examples below.

4.1.1 Example: Single Member Join

This example illustrates how member m3 join to access the resource and how resource access membership matrix is maintained. Member uses TGDH [13] to join to access the resource.

1) There are two resources namely R1 and R2;
2) Member m1, m2 accesses the resources R1;
3) Member m5,m6 accesses the resources R2;
4) Each member has to maintain resource access membership matrix.

Example:

1) Member m3 joins R1.
2) Member m3 wants to access R2, broadcast join request for R2 alongwith message containing it is already having membership with R1.
3) Each member in R1 and R2 notices and makes the entry '1' against the entry of m3 in resource access membership matrix.
4) Sponsor of R2 gives/broadcast blinded keys, membership details.
5) Thus m3 is made sponsor. Because it is a member of R1 and R2.
6) Member m3 joins for resource R2 and builds key graph.
7) Member m3 builds key graph as it is overlapped with R1 and R2. It also makes the entry in resource membership matrix.
8) Each Member of R1 and R2 has its own tree view.
9) Each member of R1 except m3 has the following view of Resource Access Membership Matrix. R1=$m1, m2, m3$; Rows represents members m_1, m_2, m_3; and Columns represents Resources R_1, R_2.

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \end{bmatrix}$$

It indicates that m1 represents that m1 is part of (access) of R1, m2 is part of R2 and m3 is part of R1 and R3.

10) Member m3 has the following view of Resource access membership matrix. Rows represents members m_1, m_2, m_3, m_5, m_6 and Columns represents Resources R_1, R_2.

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Member m3 has information of m5 and m6 because they are the members of resource group R2. RAM matrix maintained by m3 indicates that m1 is part of resource group R1, m2 is part of resource R2, m3 is part of resource R1 and R2, m4 is part of resource R2 and m5 is a part of resource R2.

11) Each member of R2 except m3 has the following view of Resource Membership Matrix. R1=m1, m2, m3. Rows represents members m_3, m_5, m_6 and Columns represents Resources R_1, R_2.

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

Figure 2 shows that m1, m2 are the members of resource R1 and m5, m6 are the members of resource R2. Figure 3 shows member m3 joins R1. Figure 4 shows member m3 builds key graph as it is overlapped with R1 and R2. Figure 5 shows each Member of R1 and R2 key tree view.

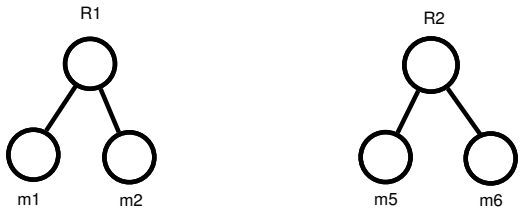

Figure 2: Members and resources

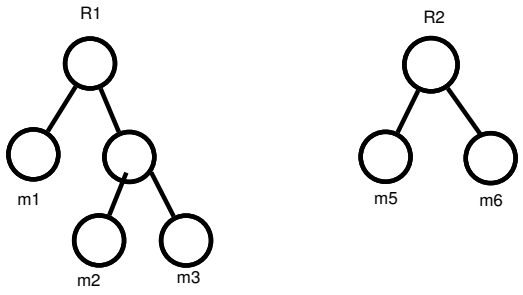

Figure 3: M3 joins R1

Table 1 illustrates the analysis of when single member joins to access the resource.

4.2 Batch Join

Buchade and Ingle states the algorithm when multiple tenants in a batch wants to access the resources [4]. The algorithm is classified into

1) Some tenants in a batch access single resource;
2) Some tenants in a batch access the multiple resources.

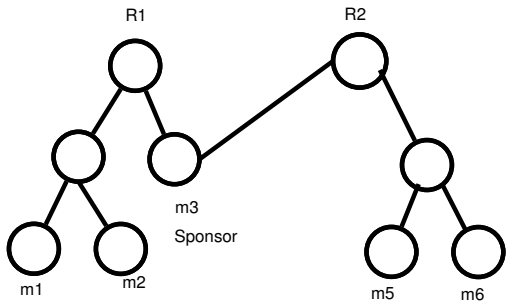

Figure 4: M3 joins R1 and R2, keygraph at M3

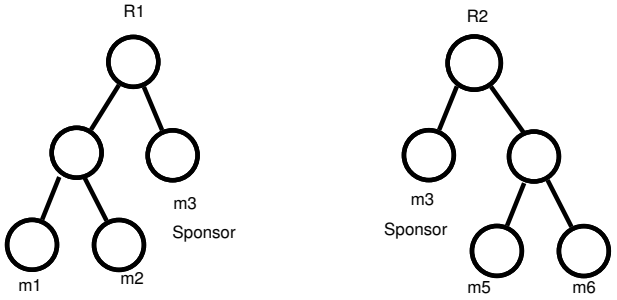

Figure 5: Each member view of R1 and R2

Key trees are build by considering overlapping members and non overlapping members in multiple resources. The details are given in the example.

4.2.1 Example: Batch Member Join

In existing key management algorithm [11, 16, 20, 24], seperate key tree is built for each resource, even if same members are accessing multiple resources.

For HDTV, Enhance layer channel subscribers can see enhance layer, Medium layer and Basic layer TV Channel. Medium layer channel subscribers can see Basic layer and Medium layer TV Channel. Basic layer channel subscribers can see Basic layer TV channel.

In existing approach, Enhanced layer members has to maintain three types of key trees.

1) For accessing EL Channel;
2) For accessing ML Channel;
3) For accessing BL Channel.

Our approach combines all key trees and eliminates redundant operations. Thus it helps to reduce key establishment time.

Each EL subscriber maintains resource access membership matrix. EL Channel considered as R1, ML Channel considered as R2 and BL Channel considered as R3. Any member broadcast request for joining the resource, the entry by EL subscriber is made in the resource membership matrix. EL subscriber forms key tree as mentioned in Figure 6.

Table 1: Analysis of single join

SE for SKT	$(N+1)(2\log(N+1)-2)$ where N = total Number of members
SE for CKT	if overlapped $(N)(2\log(N)-2)-(cm+1)(2\log(cm+1)-2)$ where cm = Number of overlapping members if not overlapped $(N+1)(2\log(N+1)-2)-(cm)(2\log(cm)-2)$

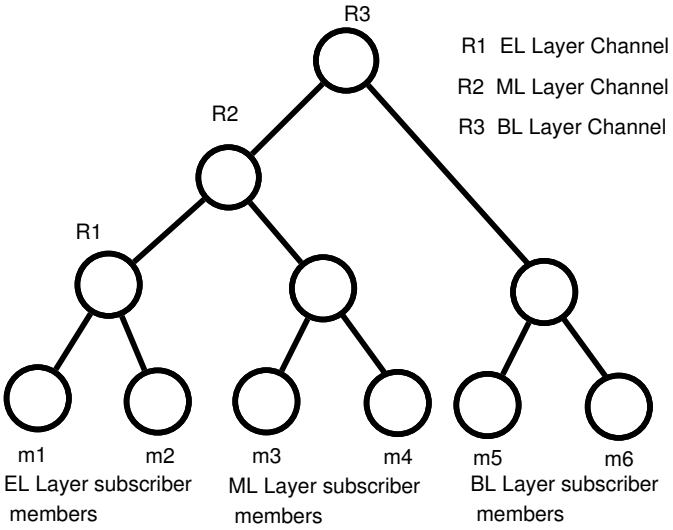

Figure 6: Key tree for EL members

EL members maintains the entries in resource access membership matrix is shown below. Members m1, m2, m3, m4, m5 and m6 represents the rows while Resources R1, R2 and R3 represents the entries in columns.

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Each ML member maintains resource access membership matrix. ML Channel considered as R2 and BL Channel considered as R3. Any member broadcast request for joining the resource, the entry by ML subscriber is made in the resource membership matrix. ML subscriber forms key tree as mentioned in Figure7.

ML members maintains the entries in resource access membership matrix is shown below. Members m3, m4, m5 and m6 represents the rows while Resources R2 and R3 represents the entries in columns respectively.

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 0 & 1 \\ 0 & 1 \end{bmatrix}$$

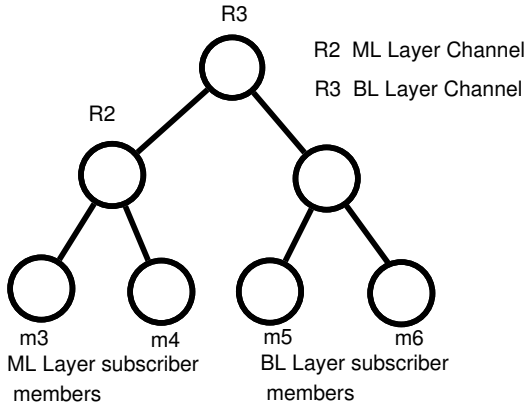

Figure 7: Key tree for ML members

Each BL member maintains resource access membership matrix. Any member broadcast request for joining the resource, the entry by BL subscriber is made in the resource membership matrix. BL subscriber forms key tree as mentioned in Figure 8.

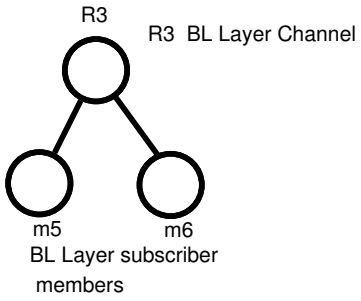

Figure 8: Key tree for BL members

BL members maintains the entries in resource access membership matrix is shown below. Members m5 and m6 represents the rows while Resources R1 represents the entries in columns.

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

Thus our approach is more suitable for the applications mentioned in [9] and reduces computation overhead in terms of sequential exponential.

Table 2 illustrates the analysis when multiple members join to access the resources.

Table 2: Analysis of batch join

SE for SKT	$(N+n)(2\log(N+n)-2)$
SE for CKT	if all subscribers overlapped $(N)(2\log(N)-2)-(cm+n)(2\log(cm+n)-2)$ if some subscribers overlapped $(N+nom)(2\log(N)-2)-(cm+om)(2\log(cm+om)-2)$ nom = Number of not overlapping members om = Number of overlapping members if not overlapped $(N+n)(2\log(N+n)-2)-(cm)(2\log(cm)-2)$

4.3 Single Leave

Buchade and Ingle states the algorithm when member leaves the access of resource [4]. It broadcast the leave request. Entry is removed from the resource access membership matrix. Each member build the key tree by considering overlapping and non overlapping members. The Table 3 gives the analysis of single leave.

4.4 Batch Leave

Buchade and Ingle states the algorithm when members in a batch leaves the access of resources [4]. The entries of the same is made in the resource membership matrix. The entries of member removed from RAM matrix when members not accessing any resources. Overlapping members builds the key tree graph and non overlapping members builds the key tree. The analysis of the batch leave is given in Table 4.

5 Results and Analysis

Analysis is done by taking resources, varying group size and overlapping members.

From Figure 9, it is observed that when number of resources are 2, Number of overlapping members, cm=30 and group size varying, Number of sequential exponentiation required for separate resource key trees required more as compared to combined resource key trees.

From Figure 10, it is observed that when number of resources are 2, group size = 200 and overlapping members varying, Number of sequential exponentiation required for separate resource key trees required (23.66%) more as compared to combined resource key trees.

From Figure 11, it is observed that when group size = 200, overlapped members = 30 and as we varying the Number of resources, sequential exponential operations for separate resource key trees are more (11.16%) as compared to combined resource key trees.

From Figure 12, it is observed that when Number of resources are 2, overlapping members = 50 and group size varying, key establishment time required more for separate resource key trees as compared to combined resource key trees.

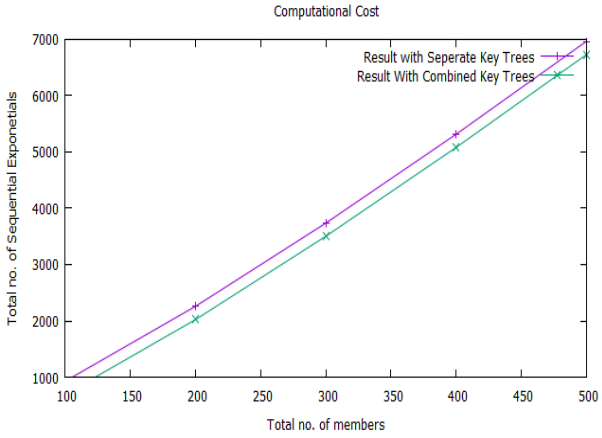

Figure 9: Computational cost, Number of resources = 2, cm = 30

From Figure 13, it is observed that when Number of resources are 2, group size = 100 and overlapping members varying key established time required for separate resource key trees required more as compared to combined resource key trees.

From Figure 14, it is observed that when Group size = 200, overlapped members = 50 and as we varying the Number of resources, key establishment time for separate resource key trees required more as compared to combined resource key trees.

6 Related Work

[14] proposes the scheme of tree key graph design but it has computation overhead for connection network generation. [2, 19] proposes share based key management scheme. KDC Scheme is used. It can cause single point of failure. Key-user tree is proposed. Storage cost is analyzed. Scheme is applicable to group communication. [8] proposes IGK scheme, considers TGDH approach. Author describes service group containing equal Number of members. But in real scenario, members can vary in the group. Sponsor selection is as per TGDH. The author ap-

Table 3: Analysis of single leave

SE for SKT	$(N-1)(2\log(N-1)-2)$
SE for CKT	if overlapping member leaves $(N)(2\log(N)-2)-(cm-1)(2\log(cm-1)-2)$ if non overlapped member leaves $(N-1)(2\log(N-1)-2)-(cm)(2\log(cm)-2)$

Table 4: Analysis of batch leave

SE for SKT	$(N-n)(2\log(N-n)-2)$ where N = Number of members
SE for CKT	if overlapping member leaves $(N)(2\log(N)-2)-(cm-om)(2\log(cm-om)-2)$ if non overlapped member leaves $(N-nom)(2\log(N-nom)-2)-(cm)(2\log(cm)-2)$ if non overlapping and overlapping member leaves $(N-nom)(2\log(N-nom)-2)-(cm-om)(2\log(cm-om)-2)$ where nom = non overlapping members and om = overlapping members

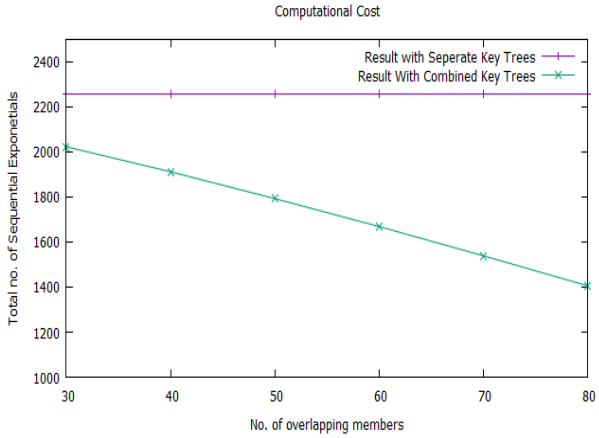

Figure 10: Average computational cost, Number of resources = 2, group size = 200

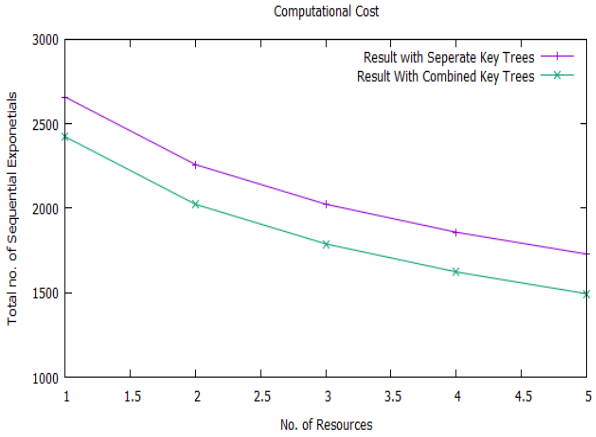

Figure 11: Average computational cost, group size = 200, overlapping members = 30

plies the scheme to specific type of example. [10] proposes tunable group key agreement protocol. Tree structure is used to form the group key among the members. [5] proposes share based hierarchical access control scheme is used. Group manager is considered. It assigns secret shares. Multi-group key management scheme is proposed. Computational analysis not done. [9] shows study of existing access control models done. Detailed analysis not done. [18] describes that group members are arranged in the hierarchical fashion. DH key agreement is applied. Sponsor not broadcasting blinded keys. But overlapping members not considered.

[16, 17] described group key formation techniques. It allows group members to consent on a shared group key. It is used to protect a shared file system present in the cloud. Any member can be sponsor. Concept of key lock boxes are used and represented in tree manner. Multiple members overlapping among different resources (e.g. files) in terms of group key management not considered. [7, 15, 22, 24, 25] uses TGDH but does not addresses issues of overlapping members. In [7] Huffman-based join-exit-tree scheme for contributory key management is proposed. It mainly concerns with key establishment time. But it does not concern with overlapping members. [3] describes key management methods and how it can be applied to computing scenario. Group key management method is also mentioned but not in detail. [11] Decisional square Diffie hellman approach is used. [21] proposes group key agreement protocol. Computational Diffie-Hellman used. But does not consider overlapping members.

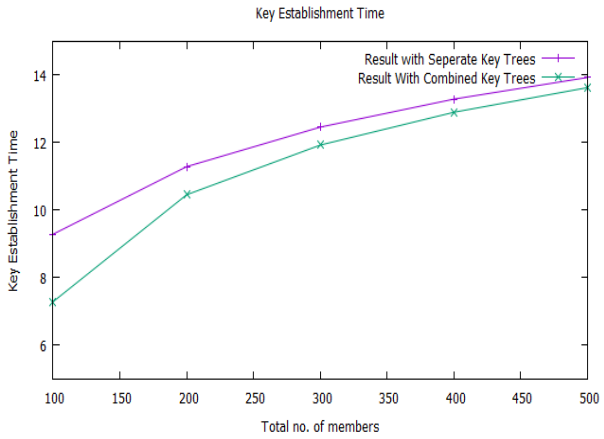

Figure 12: Key establishment time, Number of resources = 2, cm = 50

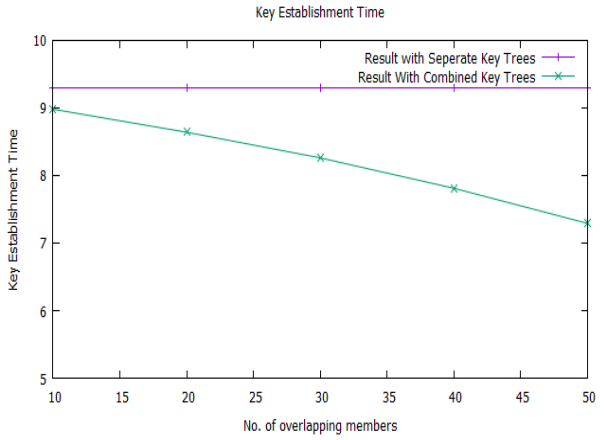

Figure 13: Key establishment time, Number of resources = 2, group size = 100

7 Conclusions

Group key is used to secure the access of resource in Cloud Computing. Group key is formed by tenants using resource key tree. TGDH is used to form the group key by building the key tree. In existing scenario, different key trees are formed even if tenants are common in multiple groups to access the resources. It causes computational overhead. We have proposed advance TGDH in which key trees may be combined if there are overlapping members in groups. Examples and analysis of algorithms are given. Through the analysis it is observed that computational overhead with respect to sequential exponentiation operations is decreased by 24% if we combine the key trees than the separate key trees. It is also observed that key establishment time for combined key trees is less compared to separate key trees.

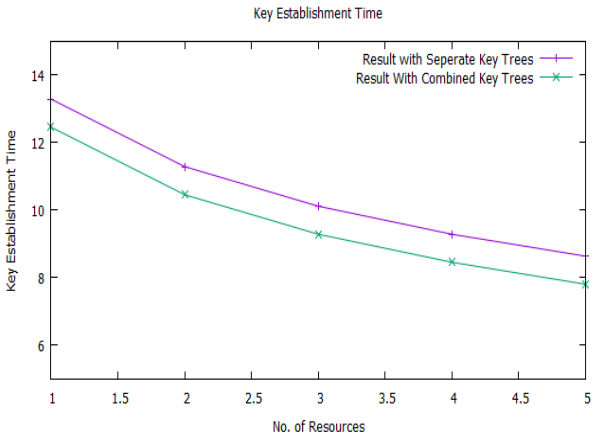

Figure 14: Key establishment time, group size = 200, overlapping members = 50

References

[1] J. Alves-Foss, "An efficient secure authenticated group key exchange algorithm for large and dynamic groups," in *In Proceedings of the 23rd National Information Systems Security Conference*, pp. 254–266, 2000.

[2] R. Aparna and B. B. Amberker, "Key management scheme for multiple simultaneous secure group communication," in *IEEE International Conference on Internet Multimedia Services Architecture and Applications*, pp. 1–6, 2009.

[3] A. Buchade and R. Ingle, "Key management for cloud data storage: Methods and comparisons," in *IEEE Fourth International Conference on Advanced Computing & Communication Technologies*, pp. 263–270, 2014.

[4] A. Buchade and R. Ingle, "Key trees combining algorithm for overlapping resource access members," *International Journal of Network Security*, vol. 18, no. 5, pp. 855–860, 2016.

[5] S. D. Dexter, R. Belostotskiy and A. M. Eskicioglu, "Multilayer multicast key management with threshold cryptography," in *Electronic Imaging*, pp. 705–715, International Society for Optics and Photonics, 2004.

[6] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[7] X. Gu, J. Yang, J. Lan and Z. Cao, "Huffman-based join-exit-tree scheme for contributory key management," *Computers & Security*, vol. 28, no. 1, pp. 29–39, 2009.

[8] X. Gu, Y. Zhao and J. Yang, "Reducing rekeying time using an integrated group key agreement scheme," *Journal of Communications and Networks*, vol. 14, no. 4, pp. 418–428, 2012.

[9] H. R. Hassen, A. Bouabdallah, H. Bettahar and Y. Challal, "Key management for content access con-

trol in a hierarchy," *Computer Networks*, vol. 51, no. 11, pp. 3197–3219, 2007.

[10] R. Ingle and G. Sivakumar, "Tunable group key agreement," in *32nd IEEE Conference on Local Computer Networks*, pp. 1017–1024, 2007.

[11] S. Jarecki, J. Kim and G. Tsudik, "Flexible robust group key agreement," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 5, pp. 879–886, 2011.

[12] D. H. Je, J. S. Lee, Y. Park and S. W. Seo, "Computation-and-storage-efficient key tree management protocol for secure multicast communications," *Computer Communications*, vol. 33, no. 2, pp. 136–148, 2010.

[13] Y. Kim, A. Perrig and G. Tsudik, "Tree-based group key agreement," *ACM Transactions on Information and System Security*, vol. 7, no. 1, pp. 60–96, 2004.

[14] H. S. Koo, O. Kwon, S. W. Ra, et al. "A tree key graph design scheme for hierarchical multi-group access control," *IEEE Communications Letters*, vol. 13, no. 11, pp. 874–876, 2009.

[15] K. Kumar, V. Sumathy, et al. "A novel approach towards cost effective region-based group key agreement protocol for secure group communication," *arXiv preprint arXiv: 1007.0087*, 2010.

[16] I. Lam, S. Szebeni and L. Buttyán, "Invitation-oriented tgdh: Key management for dynamic groups in an asynchronous communication model," in *IEEE 41st International Conference on Parallel Processing Workshops*, pp. 269–276, 2012.

[17] I. Lam, S. Szebeni and L. Buttyán, "Tresorium: cryptographic file system for dynamic groups over untrusted cloud storage," in *IEEE 41st International Conference on Parallel Processing Workshops*, pp. 296–303, 2012.

[18] S. A. Mortazavi, A. N. Pour and T. Kato, "An efficient distributed group key management using hierarchical approach with diffie-hellman and symmetric algorithm: Dhsa," in *IEEE International Symposium on Computer Networks and Distributed Systems*, pp. 49–54, 2011.

[19] B. R. Purushothama and B. B. Amberker, "Group key management scheme for simultaneous multiple groups with overlapped membership," in *IEEE Third International Conference on Communication Systems and Networks*, pp. 1–10, 2011.

[20] M. Rajaram and D. Thilagavathy, "An interval based contributory key agreement," in *IEEE International Conference on Wireless Communication and Sensor Computing*, pp. 1–6, 2010.

[21] R. S. Ranjani, D. L. Bhaskari and P. S. Avadhani, "An extended identity based authenticated asymmetric group key agreement protocol," *International Journal of Network Security*, vol. 17, no. 5, pp. 510–516, 2015.

[22] Y. Sun and K. J. Liu, "Hierarchical group access control for secure multicast communications," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1514–1526, 2007.

[23] G. Wang, O. Jie, H. H. Chen and M. Guo, "Efficient group key management for multi-privileged groups," *Computer Communications*, vol. 30, no. 11, pp. 2497–2509, 2007.

[24] H. Xiong, X. Zhang, W. Zhu and D. Yao, "Cloudseal: End-to-end content protection in cloud-based storage and delivery services," in *Security and Privacy in Communication Networks*, pp. 491–500, Springer, 2012.

[25] K. Xue and P. Hong, "A dynamic secure group sharing framework in public cloud computing," *IEEE Transactions on Cloud Computing*, vol. 2, no. 4, pp. 459–470, 2014.

Rajesh Ingle is adjunct professor at Department of Computer Engineering, Government College of Engineering Pune. He is professor in Department of Computer Engineering, Pune Institute of Computer Technology, Pune. He has received Ph.D. CSE from Department of Computer Science and Engineering, Indian Institute of Technology Bombay, Powai. He has received the B.E. Computer Engineering from Savitribai Phule University of Pune, and M.E. Computer Engineering from Government College of Engineering, Savitribai Phule Pune University. He has also received M.S. Software Systems from BITS, Pilani, India. He is a senior member of the IEEE, IEEE Communications Society, and IEEE Computer Society. His research area is distributed system security, grid middleware, cloud security, multimedia networks and spontaneously networked environments.

Amar Buchade is research scholar at College of Engineering, Pune. He has received B.E. and M.E. in Computer Engineering from Walchand College of Engineering, Sangli in 2002 and 2005 respectively. His research area is distributed system, cloud computing and security.