AN APPROACH FOR BROADER RANGE OF SECURITY IN CLOUD COMPUTING

R.Bala chandar[1], M.S.Kavitha[2], K.Seenivasan[3]

[1]PG Scholar, Department of Computer Science Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore - 641062, TamilNadu, India
balachandar.raju@gmail.com

[2]PG Scholar, Department of Computer Science Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore - 641062, TamilNadu, India
kaviktg@gmail.com

[3]PG Scholar, Department of Computer Science Engineering, Sri Shakthi Institute of Engineering and Technology, Coimbatore - 641062, TamilNadu, India
srinivassvks@gmail.com

Abstract

Cloud computing is one of the today's most exciting technologies due to its abilities like ensuring scalable services, reducing the burden of local hardware and software management associated with computing while increasing flexibility and scalability. A major feature of the cloud services is that user's data are usually processed remotely in unknown machines. Though there are many conveniences brought by this new technology, there are some issues such as cloud computing servers can be easily compromised, data is no longer under the control of the data owner and cloud computing is not responsible for the access control of outsourced data desired by the data owner. To address these problems, we propose a new framework known as Secure Flexible Framework (SFF). This proposed framework allows data owner to achieve cloud data integrity, actual data usage tracking and fine grained access control of outsourced data. Through this paper, we strengthen the correctness and user's control of their own data in the cloud along with scalable, flexible and fine grained access control of outsourced data.

Keywords

Data security, Data correctness, Information accountability, Access control, Data integrity

1. Introduction

Cloud computing is a promising paradigm that enables cloud customers to enjoy the on demand high quality applications and services from a shared pool of configurable computing resources through storing their data remotely in the cloud. This technology enabled services to be consumed as per pay and use model. This new technology has generated significant interest in the market place and organizations resulting in a number of commercial and individual cloud computing services, e.g. Amazon [15], Google[16][17], Microsoft, Yahoo and Sales force [14]. Cloud computing is a development of parallel computing, distributed computing, utility computing, virtualization and service oriented architecture. Different service oriented cloud computing models have been proposed which includes Software as a Service (SaaS), Infrastructure as a Service (IaaS) and Platform as a Service (PaaS). The cloud offerings can generally be sorted into several categories including Disaster Recovery, Application Programming Interface, Managed Services, Large Data Set Processing and Cloud Integration. In

general, cloud computing may have several customers such as non-technical users and organizations who have many different intentions to move to cloud.
In cloud computing technology there are set of important policy issues, which include issues of privacy, security, anonymity, government surveillance, reliability and liability among others. But the most important is security and how cloud providers assure it. One of the important cloud security concerns is data security and privacy because of its Internet based storage and management.

While enjoying the provision of huge amount of storage space and customizable computing resources, the computing platform shift also eliminates the responsibility of local machines for data maintenance at the same time. Although the cloud infrastructures are much more powerful and reliable than personal computing devices, broad range of both internal and external threats for data integrity such as outages and data loss incidents still exist.
In the mean while users also start worrying about losing control of their own data. The outsourced data on the cloud leads to a number of issues related to information accountability. Such fears are becoming a significant barrier to the wide adoption of cloud services.
Data is a very intense asset for any organization, and enterprise users will face serious consequences if their secret data is allowed to be seen by public and other business challengers. Thus cloud users want to make sure that their data are kept confidential to outsiders, including the cloud provider and their potential competitors.
In this paper, we propose the Secure Flexible Framework (SFF), to address the above problems which help to ensure the data integrity and information accountability in cloud with flexible access control of the outsourced data.

2. Literature survey

[1]Cloud storage provides some benefits such as reducing the burden of local data storage and maintenance by storing the user's data remotely and provides a service of utilizing the computing resources which is in a shared pool. [2]Security and reliability have been the major issues regarding the storage server. The main idea is to verify the storage server for checking the integrity of the data which is stored in the cloud. [3] To make storage services accountable for data loss, we present protocols that allow a third party auditor to periodically verify the data stored by a service and assist in returning the data intact to the customer. In particular, we consider the task of allowing a third party auditor (TPA), on behalf of the cloud client, to verify the integrity of the dynamic data stored in the cloud. [4]Enabling public auditability is important so that users can resort to a third party auditor (TPA) to check the integrity of outsourced data. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities towards user data privacy, and introduce no additional online burden to user.[5] It describes three key problems for trust management in federated systems and presents a layered architecture for addressing them. Three problems are addressed which includes the trust verification, monitoring and re evaluating the trust relationship based on historical past behaviour. [6] A language is proposed that allows data distribution with usage policies in a decentralized architecture. Logic is designed that allows audited agents to prove their actions and authorization to posses' particular data. Accountability is defined in several flavors, including agent accountability and data accountability. Finally, they show the soundness of the logic. [7] Here it analyses the how accountability is transferred by the delegator when he/she is transferred some of their right to the delegate.[8] It introduces a model for provable data possession (PDP).It stored data at an untrusted server to verify that the server possesses the original data without retrieving it Thus, the PDP model for remote data checking supports large data sets in widely-distributed storage systems. In particular, the overhead at the server is low (or even constant), as opposed to linear in the size of the data.[9] They describe an operational model of accountability-based distributed systems. We describe analyses which support both the

accountability design systems and the auditors validation for finite accountability systems. Our study provides the power of the auditor, the efficiency of the audit protocol, the requirements placed on the agents, and the requirements placed on the communication infrastructure.[10] In this Paper they present a system for realizing complex access control on encrypted data that we call Ciphertext-Policy Attribute-Based Encryption. By using this technique encrypted data can be kept confidential even if the storage server is untrusted; moreover, this method is secure against collusion attacks. Here the attributes are used to describe a user's credentials, and a party encrypting data determines a policy for the user who can decrypt. Thus, this method is conceptually closer to traditional access control methods such as Role-Based Access Control (RBAC).[11]This approach is used to keep sensitive user data confidential against untrusted servers; existing solutions usually apply cryptographic methods by disclosing data decryption keys only to authorized users. Here they defining and enforcing access policies based on data attributes, and allowing the data owner to delegate most of the computation tasks involved in fine-grained data access control to untrusted cloud servers without disclosing the underlying data contents by combining techniques of attribute-based encryption (ABE), proxy re-encryption, and lazy re-encryption.[12]They develop a new cryptosystem for fine-grained sharing of encrypted data that we call Key-Policy Attribute-Based Encryption (KP-ABE).It construct a model to supports delegation of private keys which subsumes Hierarchical Identity-Based Encryption (HIBE).[13] Attribute-Based Encryption (ABE) is a new paradigm where the policies are specified and cryptographically enforced in the encryption algorithm itself. In this work they focused on improving the flexibility of representing user attributes in keys. Specifically, they proposed Ciphertext Policy Attribute Set Based Encryption (CP-ASBE).

3. Problem Statement

3.1 Components

Various network entities are as follows:

- **User**: It is an entity which can be an enterprise or an individual who depends on the cloud for data storage and computation.
- **Cloud Server**: It is an entity that can be managed by cloud service provider to offer data storage service and has significant storage space and computation resources.
- **Third party Auditor**: It is an optional trusted entity to assess and expose the risk of cloud storage services based on users request.
- **Logger**: An entity, which is responsible for automatic logging access to data items and for generating error correcting information for each and every logged activity.
- **Logging Auditor**: An entity, which is responsible for auditing the logged information.
- **Trusted Authority**: An entity, which is the root authority for managing top level domain authorities.
- **Domain Authority**: An entity, who administrates the data owner/consumer.

3.2 System Model

The System model is illustrated in Fig 1.When the data owner wants to store their data items into the cloud; he initially tokenizes the data items depending upon the storage capacity of the

cloud servers. Next, for each and every token, the digital signature is generated by the data owner. Then the encrypted tokens are stored randomly in a distributed manner into the cloud servers. During the retrieval time of the data items, all the encrypted tokens are integrated together to form the original data item.

With the original data item, a logger and some access policies are combined to form the jar file. The jar file includes a set of simple access control rules specifying whether and how the cloud servers and possibly other stakeholders are authorized to access the content itself.

The trusted authority of the organization has the rights to receive the jar file and make them available to the domain authorities. The domain authorities are under the control of trusted authority. The main responsibility of the domain authority is to authorize and manage the data consumers.

3.3 Design Goals

To ensure the security, we aim to design efficient mechanisms and achieve the following goals:

1. **Storage correctness**: To ensure that the users data which is stored in the cloud is kept intact all the time.
2. **Data error localization**: To effectively identify the misbehaving server when data corruption has been detected.
3. **Dynamic data support**: To assure the same level of storage correctness even if users edit their data files through modifying, deleting, or appending.
4. **Distributed logging:** Decentralization of logging should be achieved in order to adapt the dynamic nature of cloud.
5. **Automatic logging**: The automatic logging should be made correctly in every access of the user's data.
6. **Periodic push and pull**: To inform about the current usage of data, periodical logging of files must be sent back to the data owners.
7. **Fine grained access control**: Each and every level of users attribute should have a set of separate keys.
8. **Scalability and Flexibility**: To achieve an efficient access control, we should have the flexibility in attribute management and scalability in dealing with multiple levels of attribute authorities.

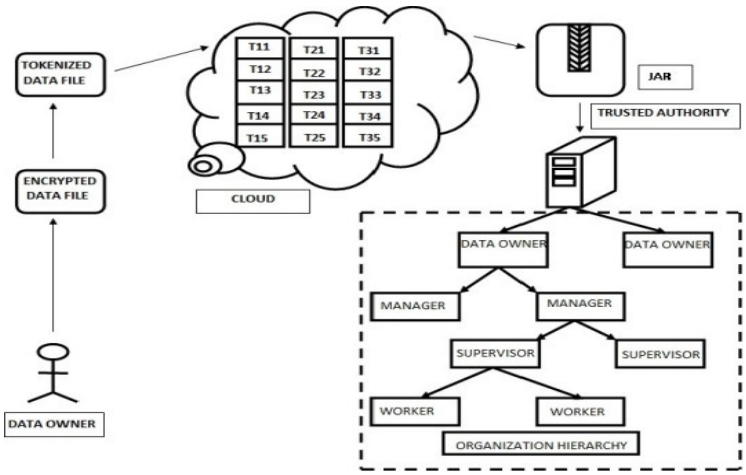

Figure 1 - System Model

4.0 Secure Flexible Framework

In this section, we present an overview of the Secure Flexible Framework and discuss how the SFF meets the design requirements discussed in the previous section.
The Secure Flexible Framework proposed in this work has various automatic logging and decentralized auditing of appropriate access performed by any entity, which can be done at any point of time within any cloud service provider. In addition to that, we create an effective storage verification scheme for dynamic data support to ensure the storage correctness of data with fine grained access control of outsourced data in the cloud with various levels of users in an organization.

4.1 MECHANISMS

4.1.1. Erasure correcting code

Erasure correcting code is used to distribute the file with replication to guarantee the data dependability. It reduces the communication and storage overhead as compared with the traditional replication-based file distribution techniques.

4.1.2 Homomorphic tokenization

By utilizing the homomorphic tokenization with distributed verification of erasure-coded data, SFF achieves the storage correctness and data error localization. The storage correctness is verified along with the detection of data corruption. This helps to identify the compromised server(s).

4.1.3 Logging (logger)

By adding it with the users' data items, it can be downloaded when the data is accessed and whenever the data is copied.

4.1.4 Auditing

This is processed with the help of the logged files provided by the logger. The following are the two modes of operations:

1. **Shove –in:** The log files are shoved back to the data owner periodically in an automated fashion

2. **Fling – Out:** It is an on demand approach, where the log files are obtained by the data owner as often as required. It is also responsible for handling log file corruption

4.1.5 Attribute based encryption

Attribute based encryption scheme will be used for encrypting both the cipher text and users' decryption keys that are associated with a set of attributes or policy over attributes. Decrypting a cipher text is done only if the decryption key and the cipher text have a match.

4.2. DATA FLOW

The overall Secure Flexible Framework combines the data items, users, cloud storage, loggers, logger auditor, tokenization process and how ABE scheme is applied in the organization (with various levels of users) is sketched in Fig 2. At the beginning, the client login into the cloud

server through their user name and password based on the Identity Based Encryption (IBE) scheme. The uploading file is obtained from the data owner only after successful authorized access verification. Then the original file is tokenized into equal size streams of tokens and stored into the same size of blocks at various cloud servers randomly with some access control which is desired by the data owner.

When the data is accessed from the cloud server, the streams of tokens are merged to form the original file. Before that, the originality of the file content will be verified through the digital signature that is generated for each stream of tokenized files which are in the cloud servers. If any intruder tries to modify the tokenized file, we can easily identify the compromised server since the digital signatures are created for every stream of tokens.

After the rearranging process, the original file is added with the logger to form the JAR file with some access policies. This logger file contains the details about the data items which are accessed by the stakeholder or organization. As for the logging, whenever there is an access to the data, the JAR will automatically generate a log record, encrypt it using the public key which is distributed by the data owner, and store it along with the data. The encryption of the log file prevents the unauthorized changes created by the attackers to the log file. The data owner could opt to reuse the same key pair for all JARs or create different key pairs for separate JARs. If any of the unauthorized users or any cloud service provider is trying to misuse the data items, it will be easily identified by giving an alert to the data owner.

Finally, the trusted domain of this framework will read the JAR and generate a master key for all the domain authorities those who are in need to access the outsourced data. After the data is outsourced from the cloud, the privileged domain authorities will generate the keys for every attribute at each level of users' in the organization.

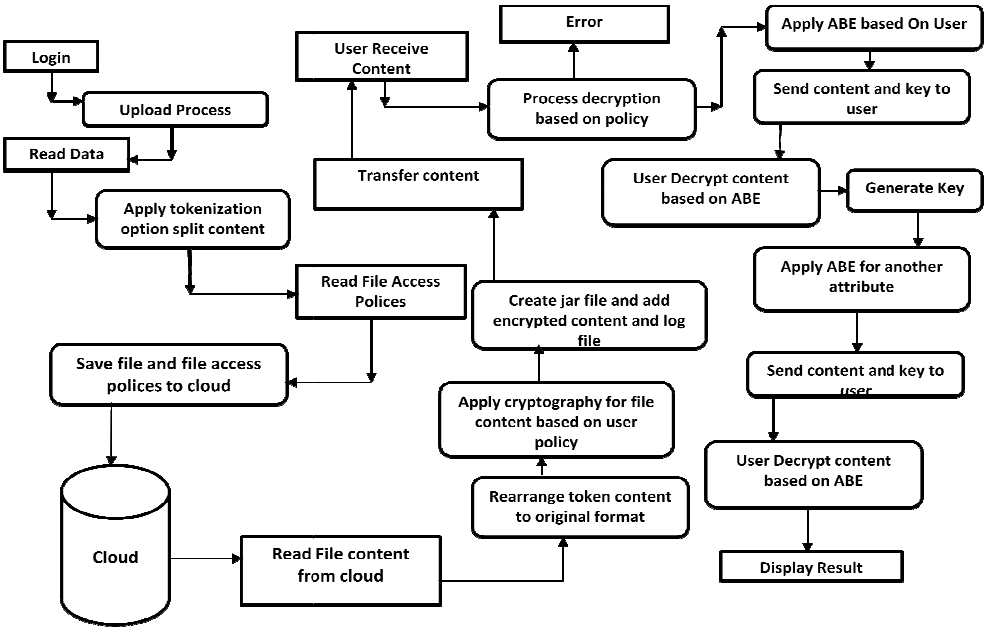

Figure 2 - Data flow in SFF

The domain authority/authorities are under the control of the trusted authority, so it can generate the key for the next level of users. So the burden of the trusted authority for managing and generating keys will be reduced as well as the scalability and flexibility via efficient attribute set management will be improved. Each level of users has their separate key for each of their attributes, so fine grained access control of the outsourced data is possible with this framework

5. IMPLEMENTATION

We begin this section by considering an illustrative example which serves as the basis of our implementation and demonstrate the main features of our system.
Example: Consider ABC as a jewellery shop with some levels of users as shown in Fig 3, plans to store their jewel reports into the cloud and to access the same with the following requirements:

1. The jewel reports are downloaded only by the authorized users who are in that shop.
2. Users in this shop should be equipped with security means so that they can make continuous correctness assurance of their jewel report even without the existence of local copies.
3. The online burden of data integrity checking tasks must be reduced in an efficient manner
4. The cloud provider or stake holder should not misuse the data for their own purpose without the knowledge of data owner
5. The outsourced data from the cloud should be accessed only by the authorized domain authorities (ABC Jewel shop).
6. Each user at various levels in the shop must have limited access to the outsourced data, as desired by the data owner or higher level entities in the organization.

In order to achieve the above requirements, we have implemented a private cloud using Java/J2EE and MySql environment (Eclipse and Netbeans).According to Fig 3, only the admin or data owner has the authority to store the jewel report into the cloud. Before storing the jewel report into the cloud, the report will be tokenized using homomorphic tokenization mechanism. The number of tokens depends on the cloud servers. For each and every token, digital signature is generated using RSA algorithm.

Then the digitally signatured tokens are maintained locally in order to assure the data integrity. All the signatured tokens are placed randomly across the servers. Here the trusted authority behaves as an intermediator between the server (cloud service provider) and the client (admin or data owner). When the trusted authority tries to access the jewel report, initially all the signatured tokens in the cloud servers are verified with the locally maintained digital signature tokens.

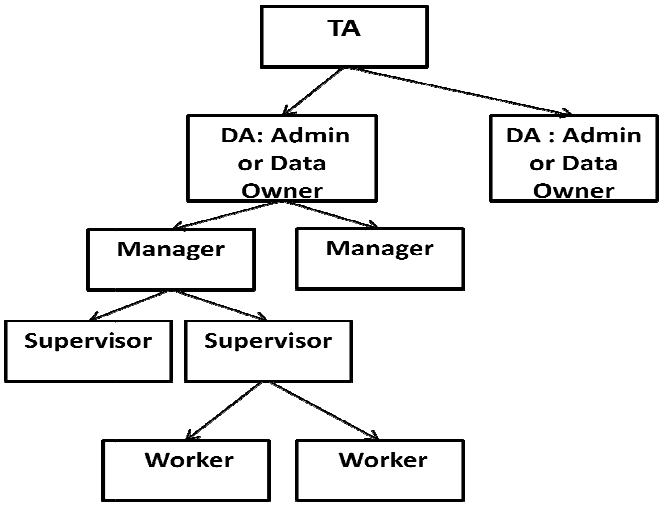

Figure 3 - Entity levels in an organization

After the verification, all the tokens are integrated together to form the original jewel report. The verification process helps to find the compromised servers in case if their corresponding tokens are modified.

Through the above procedure we can achieve the second requirement. To the original jewel report, a logger file with some access policies is added in order to form the jar file. Now the jar file behaves as a container which consists the logger file in an encrypted format and the access policies which are enforced by the data owner. The logger file contains the accessed information of the jewel report such as name, time, location, type of access of the user. The logger file automatically saves the details when the jewel report is accessed by the user. Periodic auditing is processed by the logger auditor to identify the unauthorized access. The periodic auditing process has two separate modes such as shove-in and fling-out.

During the shove –in mode, the log files are shoved back to the data owner periodically in an automated fashion. Fling-out is an on demand approach, where the log files are obtained by the data owner whenever it is required. End to end information accountability is achieved along with the identification of the misusage of original jewel report. This helps to satisfy the first and fourth requirements. In order to reduce the online burden of the user, we implement the third party auditing to conduct the data integrity checking process. Using third party auditor, we can achieve the third requirement.

To satisfy the fifth and sixth requirements, the higher level entities are responsible for generating the keys for the lower level entities in the jewellery shop using Attribute Based Encryption scheme. According to the access policies defined by the data owner, some details of the jewel are hidden to the lower level entities by the higher level entities as shown in figure 4.This scheme improves the scalability, flexibility and fine grained access control of the jewel report in the cloud.

Number of Levels In the Organization	Attributes / Users	Jewel Name	Dealer Name	Buying Date	Total Grams	Sales Rate	Carat Type	Buying Date	Original Data File: Jewel Image And Description	
Level 1	Administrator or Data Owner	YES	YES	YES	YES	YES	YES	YES	YES	YES
Level 2	Manager	YES	YES	*NO*	YES	YES	YES	YES	YES	YES
Level 3	Supervisor	YES	*NO*	*NO*	YES	YES	YES	YES	YES	YES
Level 4	Worker	YES	*NO*	*NO*	YES	YES	YES	*NO*	YES	YES

YES – Permission granted for access the attributes NO- Permission not granted for access the attributes

Figure 4 - Access control of entity levels in an organization based on ABE scheme

6. Results

The following graphs are showing the result analysis of SFF using various parameters. The Graphs are shown in the figure 5-9.In figure 5, the auditing time is taken as a metric. The auditing time is defined as the time taken for verifying the received tokens with the locally computed tokens. The auditing task is delegated to the third party auditor (TPA) in order to reduce the online burden. This graph shows the achievement of time reduction during auditing by third party auditor when compared with the data owner.

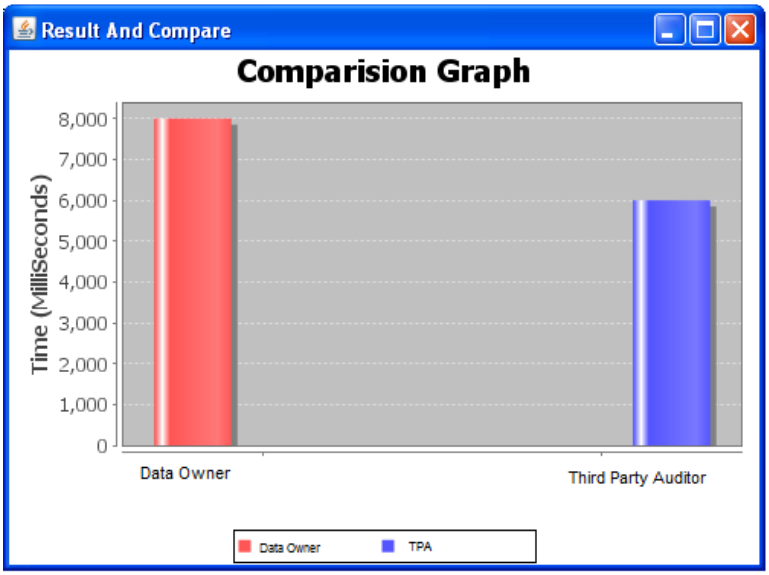

Figure 5 - Auditing Time Taken By Data Owner and TPA

In figure 6, the file size is taken as a metric. As the file size varies, the total taken for tokenization also varies.

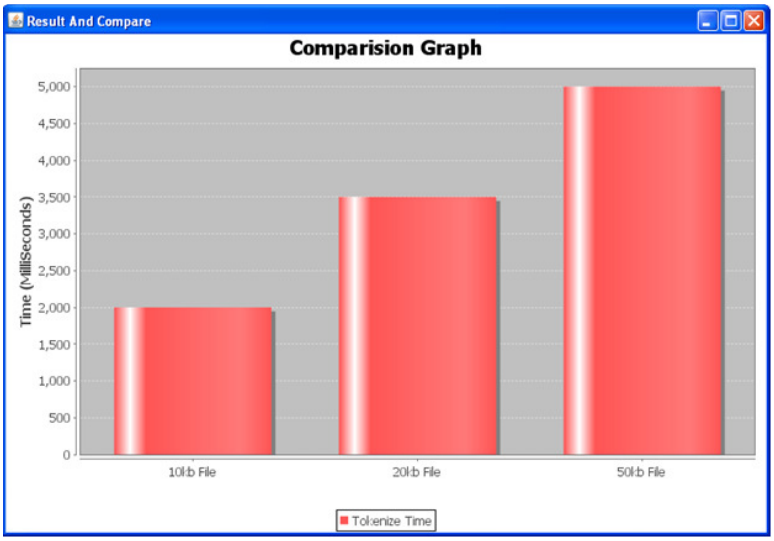

Figure 6 - Tokenization of File in Various Sized files

In figure 7, the integration time is considered as a metric. The integration time specifies the total time required for the jar creation. JAR creation is a process of combining the logger and access polices with the original data items.

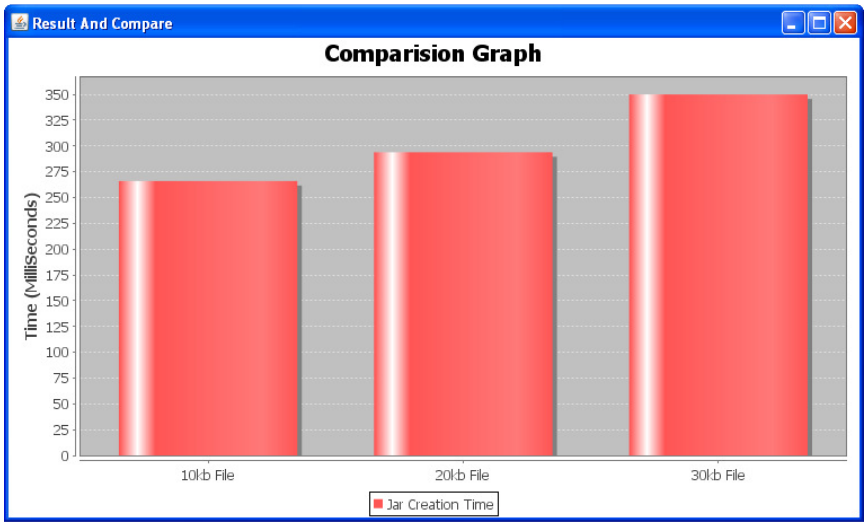

Figure 7 - JAR Creation Various Sized Files

In figure 8, the merging time is used as a metric for measuring the time taken for merging all the loggers available at various entitites in the cloud system.The distributed system(Red curve),takes less time for merging and its stable at a point when compared with the centralized aaproach(Blue curve) where the curve is linearly decreasing when the file size increases.

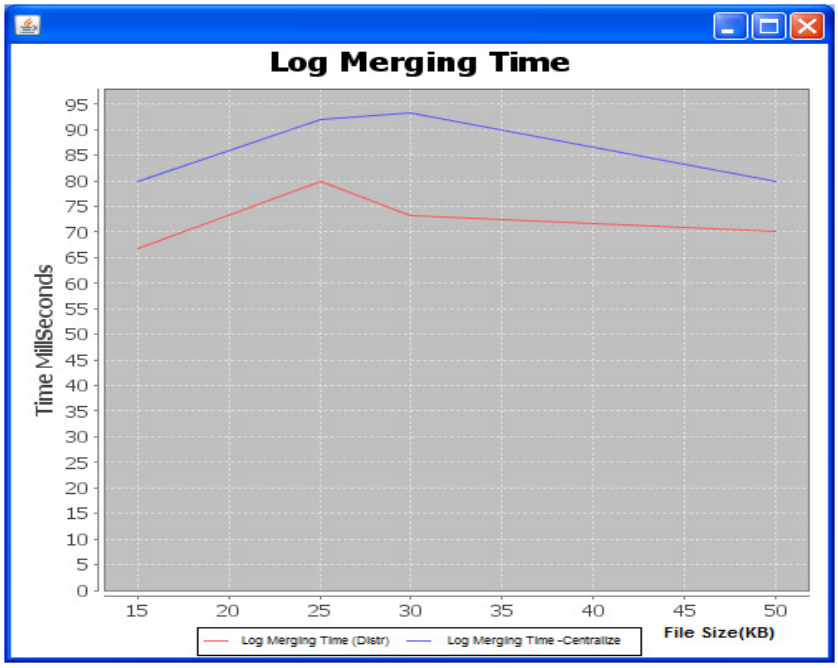

Figure 8 - Log Files Merging Time

In Figure 9, the key cost is considered as a metric. The graph shows that the time to create a key for each attribute of a user in an organization is more by using the Hierarchical attribute based encryption scheme. But in this approach, it reduces the key cost by decreasing the number of key generations for each user in an organization. This will not affect the scalability and flexibility of the system.

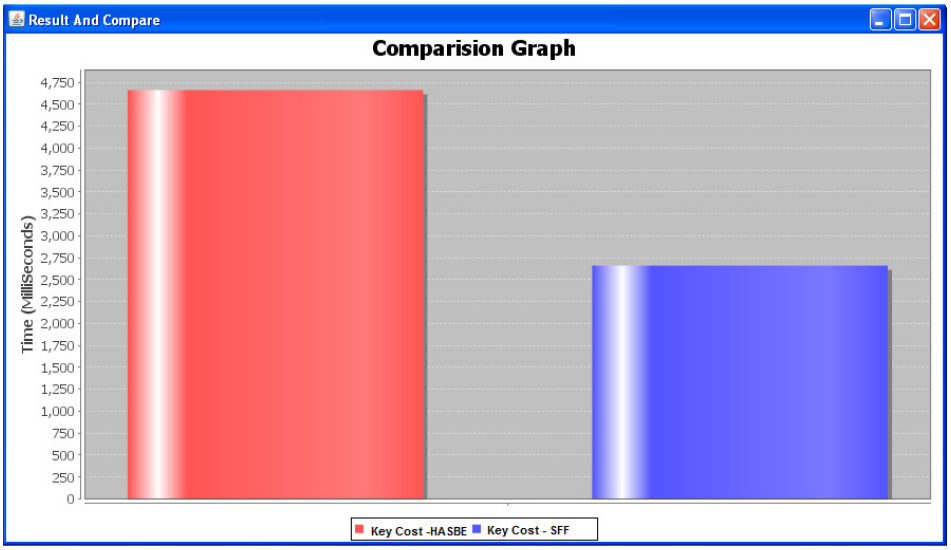

Figure 9 – Key Generation Time of HASBE and SFF

7 .CONCLUSION

Secure Flexible Framework (SFF) is proposed to achieve a broader range of security from the data owner to the data user which improves the flexibility, scalability and fine grained access control of data. It also provides the security over the data in transit and identify if any server misbehaves.

7. References

[1] C. Wang, Q. Wang, K. Ren, and W. Lou, “Privacy-Preserving Public Auditing for Data Storage ecurity in Cloud Computing,” Proc. IEEE 29th Int’l Conf. Computer Comm. (INFOCOM), pp. 525- 533, 2010.

[2] G. Ateniese, R.D. Pietro, L.V. Mancini, and G. Tsudik, “Scalable and Efficient Provable Data Possession,” Proc. Fourth Int’l Conf. Security and Privacy in Comm. Netowrks (SecureComm), pp. 1-10,

2008.

[3] M.A. Shah, R. Swaminathan, and M. Baker, “Privacy-Preserving Audit and Extraction of Digital Contents,” Cryptology ePrint Archive, Report 2008/186, http://eprint.iacr.org, 2008.

[4] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, “Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing,” Proc. 14th European Conf. Research in Computer Security (ESORICS ’09), pp. 355-370, 2009.

[5] B. Chun and A.C. Bavier, “Decentralized Trust Management and Accountability in Federated Systems,” Proc. Ann. Hawaii Int’l Conf.System Sciences (HICSS), 2004.

[6] R. Corin, S. Etalle, J.I. den Hartog, G. Lenzini, and I. Staicu, "A Logic for Auditing Accountability in Decentralized Systems," Proc. IFIP TC1 WG1

[7] B. Crispo and G. Ruffo, "Reasoning about Accountability within Delegation," Proc. Third Int'l Conf. Information and Comm. Security (ICICS), pp. 251-260, 2001.

[8] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," Proc. ACM Conf. Computer and Comm. Security, pp. 598-609, 2007.

[9] R. Jagadeesan, A. Jeffrey, C. Pitcher, and J. Riely, "Towards a Theory of Accountability and Audit," Proc. 14th European Conf.Research in Computer Security (ESORICS), pp. 152-167, 2009.

[10] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in Proc. IEEE Symp. Security and Privacy, Oakland, CA, 2007.

[11] S. Yu, C. Wang, K. Ren, and W. Lou, "Achiving secure, scalable, and fine-grained data access control in cloud computing," in Proc. IEEE INFOCOM 2010, 2010, pp. 534–542.

[12] V. Goyal, O. Pandey, A. Sahai, and B.Waters, "Attibute-based encryption for fine-grained access control of encrypted data," in Proc. ACM Conf. Computer and Communications Security (ACM CCS), Alexandria, VA, 2006.

[13] R. Bobba, H. Khurana, and M. Prabhakaran, "Attribute-sets: A practically motivated enhancement to attribute-based encryption," in Proc. ESORICS, Saint Malo, France, 2009.

[14] B. Barbara, "Salesforce.com: Raising the level of networking," Inf. Today, vol. 27, pp. 45–45, 2010.

[15] Amazon Web Services (AWS) [Online]. Available: https://s3.amazonaws.com/

[16] Google App Engine [Online]. Available: http://code.google.com/appengine/

[17] K. Barlow and J. Lane, "Like technology from an advanced alien culture:Google apps for education at ASU," in Proc. ACM SIGUCCSUser Services Conf., Orlando, FL, 2007.

Authors

Bala chandar.R. was born in Tamilnadu, India, in 1986. He received his Bachelor of Engineering degrein Computer Science and Engineering from Dr.Mahalingam College of engineerin g and technology,Pollachi under Anna university, **Chennai** in 2008.He is currently pursuing M. E in Computer Science and Engineering in Sri Shakthi institute of engineering and Technology, under Anna University, **Chennai,India**.

Kavitha.M.S was born in Tamilnadu, India, in 1989. She received her Bachelor of Technology degree in Information Technology from Vivekanandha Institute of Engineering and Technology for Women , Nammakkal, under Anna university, Chennai in 2010.She is currently pursuing M. E in Computer Science and Engineering in Sri Shakthi Institute of Engineering and Technology, **Coimbatore**, under Anna University, **Chennai,India**.

Seenivasan.K. was born in Tamilnadu, India, in 1989. He received his Bachelor of Engineering degree in Computer Science and Engineering from P.S.R Engineering College, Sivakasi, under Anna university, Chennai in 2010.He is currently pursuing M.E in Computer Science and Engineering in Sri Shakthi institute of engineering and Technology under Anna University, **Chennai, India.**