A FRAMEWORK ASSURING DECENTRALIZED ACCOUNTABILITY INTHE CLOUD

Thasni T[1], P. Mohamed Shameem[2]

[1]*Student, Computer and Information Science, Cochin University of Science & Technology (CUSAT) , Kerala, India*
[2]*Associate Professor, TKM Institute of Technology, Kollam, Kerala*

Abstract

Cloud computing enables on-demand and convenient network access to a shared pool of resources. One of the major characteristics of theservices in thecloud leads to an important problem in the cloud computing scenario. That is, user's data are processed on unknown remote servers which are not under the control of service owner. Multiple users want to do business of their data or application using cloud, but their fears of losing control over their data or application become an obstacle to the popularity of services of cloud. Data owners must get information that their data is not misused on the cloud. To address these problems, here proposes a decentralized framework to monitor the actual usage of the user's data available on the cloud. The proposed framework in this work performs automated logging and distributed auditing of relevant access performed by other entity, occurred at any time, at any cloud provider. The service model considered in this work is software as a service. The methodology includes an object centered approach that uses programmable capabilities of JAR to create an object that contains service owner's policies and data. When the data are accessed by any external entity, logging mechanism is triggered automatically at that moment itself. Automatic JAR checking is also done to verify the trustworthiness of the cloud server. Data encryption technique, policy checking and technique to prevent modification of JAR are the important features of this framework.

Keywords—*accountability, auditing, automatic JAR checking, cloud computing, integrity checking, logging, log records, service owner*

----------***----------

1. INTRODUCTION

Cloud computing is emerging and considered as the next generation architecture for computing. It is a combination of computing resources that can be accessed with the internet. But in cloud computing, since the data is stored anywhere across the world, the service owner has little control over their applications after uploading it in the cloud provider. Google, Amazon, Sales force and Microsoft etc are the notable commercial and individual cloud computing services. Service owner of a cloud service may not know the host systems where their data or application is being processed. To solve this problem, it is essential to provide a good mechanism for users to track the activities. On their data in the cloud. So, service owner need to be able to make sure that their data or applications are used based on the service level agreements that they have signed with the cloud provider.Cloud environments have the following properties. Cloud provider can outsource the handling of data to others in the cloud. Also entities are free to leave and join the cloud according to their wish.

Hence information handling in the cloud moves through a situationthat does not exist in conventional environments. So the approachesfor access control used in conventional environment are not suitable for the cloud. The little confidence in providing sensitive information to cloud computing service providers affects the popularity of services of cloud, as per the various reports. So, research is needed to increase the accountability, auditability of CSP to enhance trust in them [6]. To overcome the above problems, here proposes a novel framework assuring decentralized accountability in the cloud. Accountability of a cloud consists of accepting responsibility forhandling of confidential data for processing and otherwise using the data according to contractual and legal requirements from time it is collected until when the data are deleted [1]. We need to identify the need for accountability in the cloud [6].Consider a trust related scenario where a data owner or an application provider (Fig 1) shows a typical trust-related scenario which many potential cloud users worried about. A customer stores some sensitive information in a file within a virtual machine (VM) hosted by a provider she has subscribed to. After uploading the file, mechanisms for fault tolerance within the cloud will typically perform back up process, and will carry out load balancing by creating redundancies across several virtual servers and physical servers within the domain of service provider.

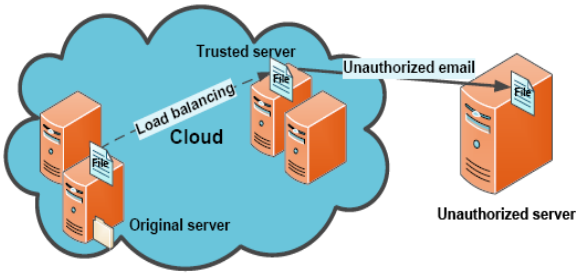

Fig.1. A situation in cloud computing environment showing the need of auditability and accountability in the cloud

These transactions and the creation of new duplicate files are needed to be monitored, accounted and logged to trace the file history and log the history of access and modifications of content, that means achieving auditability and accountability of cloud. If an entity of the cloud provider tries to send sensitive file/ data to a target outside the cloud as an unauthorized email it is very essential to know what, when,where and how was being leaked and by whom that leakage occurs.This will enhance the confidence of both CSP and the consumers. The above problem statement can be solved using the proposed framework. Automated logging and distributed auditing are the highlights of this framework. Automatic JAR checking is also done to verify the trustworthiness of the cloud server. Data encryption technique, policy checking and technique to prevent modification of JAR are the important features of this framework. Section two describes the related work and section three describes the problem statement. Proposed system is described in section four. It is followed by conclusion and references.

2. RELATED WORK

Ryan K L Kop , Bu Sung Lee, Sinai Pearson[6] highlighted accountability and auditability as an important perspective towards increasing trust in cloud computing. They proposed Cloud Accountability Life Cycle (CALC) and three abstraction layers. Tools and approaches can be designed by the researchers according to this life cycle. Marco Cassava Mont, Sinai Pearson, Pete Bram hall[2] specifically addressed two important problems: letting users be more in control of their personal data and making enterprises and organizations be more accountable of their behaviors, while dealing with users' sensitive information.

They introduced a model based on policies that are sticky, to associate tamper resistant privacy policies to obfuscated data, along with trusted tracing services.Smith Sundareswaran, Anna C. Squicciarini, Dan Lin [5] addressed the newly emerging data privacy problems in the cloud caused by indexing. They proposed a three-tier data protection framework consisting of three protection strategies which differ according to the level of privacy required by the end users. Smitha Sundareswaran, Anna C. Squicciarini, Dan Lin [8] proposed a framework called Cloud Information Accountability (CIA) framework. The CIA framework[8] provides end-to-end accountability in highly distributed fashion. The approaches of authentication, access and usage control are combined. Information accountability aims at providing transparency to the data usage. Y. Chen et al [7] presented a novel software integrity verification primitive, based on actual execution oblivious hashing computes a fingerprint of the fragment of code.

3. PROBLEM STATEMENT

A service owner or an application provider is interested in uploading a software service S on the cloud, which can be accessed by a group of external users. Significantly, that service owner does not have any physical control over the cloud machines in which his service is running, and he cannot directly observe their status (Fig. 2).Before uploading the software, the service owner and the cloud provider also enter into an agreement A that describes how the cloud is going to handle service S. Typically, A specifies at least that the cloud machines will faithfully handle the software provided by the service owner. A can also specify other constraints, such as a service level agreement, an availability objective, or a rule that the information provided to S should not be disclosed to another entity without his concern. The problem is that the service owner of S cannot ensure that S is handled by cloud provider according to the service level agreement. There is a chance that cloud provider can gain extra revenue with S by outsourcing it. It is very unwelcome to the service owner.

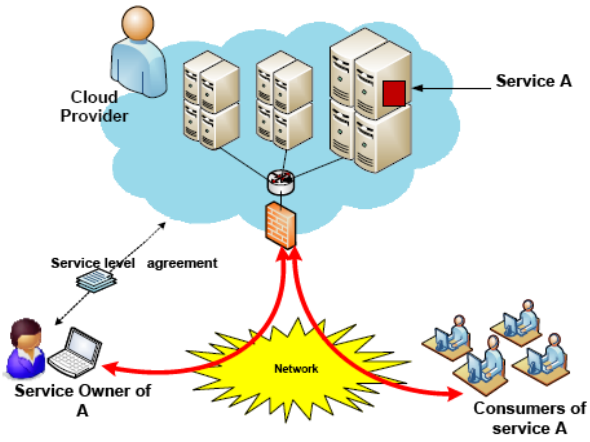

Fig 2 A scenario in cloud computing. The service owner runs a service on the cloud, but he has no control over the cloud machines

Accountability appears to be a promising approach to the problems mentioned above. Service owner of a cloud which is accountable must be able to detect whether the cloud is functioning as agreed. Suppose Alice wants to do business with some photographs using a particular cloud.

- Consumers must access the data according to their access privileges only, different access privileges must

be given to different customers based on the nature of the work.
- In any case of disputes, the corresponding service owner can have access details of a consumer.
- The service owner wants assurance that integrity of data is maintained.
- Requires distributed auditing facilities based on log records, data will comprise of all types of files, integrity of data must be maintained to avoid modifications by the consumers.
- Need automatic checking of the trustworthiness of the cloud provider to verify whether the cloud provider is working as agreed.

A service owner needs to send his data and logging policies to the cloud provider.In order to track the usage of the data, we aim to develop novel automated logging and distributed auditing techniques which satisfy the following requirements. In order to adapt to dynamic nature of cloud, the logging should be decentralized, automatic logging of every access to the data and log files should be reliable and tamper proof to avoid modification by malicious parties, log records must be provided to their data owners at equal intervals of time to make them aware of the current usage of their application. Log records must be retrievable by their data owners according to their needs.

4. PROPOSED SYSTEM

Here the proposed system is a framework assuring decentralized accountability in the cloud. It will track the usage of owner's data. This framework is developed to bring trust between cloud provider or end user and the service owner regarding the usage of data. This framework also enforces the proper handling of the service owner's data or application according to the SLA. Another advantage of this framework is that, it can track the usage of data, in future if any conflict arises then it can easily be traced down by the service owner. Wehave used the programmable capability of JAR (Java Archives) files to automatically log the usage of the service owner's data by any entity in the cloud. Service owners will forward their application and their policies for access control and logging together included in JAR file, to cloud providers.An access to the JAR file will initiate an automatic logging technique which is local to the JAR file at that moment itself. Since policy mechanism is travelling along with data it is called as strong binding. However, this binding will be there even when duplicates of the JAR file are made. So the service owner will have control over his data at any location. This decentralized type of logging mechanism is more suitable for the dynamic property of the cloud. The important feature is to log the actions that are performed on the users' data. In this system, we support three types of actions such as view, download, timed access. Here we propose different methods to record or enforce each action based on the logging method. In pure log method, every access to the data will be recorded.JAR will record the access information and the duration of access in the case of access log method. The important highlights of this framework are automatic JAR checking, policy checking without java security policy, data encryption technique and integrity checking of JAR.

4.1 Major Components and Modules

4.1.1 Service Owner Module

This module operates at the data owner. The service owner first creates and stores a JAR file. The JAR file is given a particular name. The JAR file creation also helps to define keywords, which can be used by the consumer for easily searching the programs. The JAR developed by the service owner contains data/applications along with logging mechanism and class file to maintain the integrity of the same. Data encryption technique is used to protect sensitive data in the JAR. In our work, service owner is going to do business with his photographs. An encryption algorithm is used to provide security to image files. Here rather than appending java policy to the JAR, instant policy checking algorithm is enclosed in the JAR. And it is responsible for log record generation.JAR is created along with description and executable JAR file is sent to the cloud. The structure of the log record is given below.

UR=< ID, IP adr, Act, Start Time, End Time, App name >

4.1.2 Cloud Provider Module

The cloud service provider has the responsibility to manage and control a cloud to provide services. Cloud service proider validates the providers and the consumers. It has the duty to add new users .Data owners can upload their applications in the cloud with the JAR file created for sharing with data consumers. Consumers can search for the jar files .JAR file can be downloaded by the consumers from CSP.

4.1.3 Consumer Module

The consumer is an entity that is subscribed to a particular CSP. The CSP grants access to only authenticated entities. For this purpose the users who want to access the services from the CSP must have to pay a particular amount of money to the CSP. Only authenticated entities are allowed to access the services from the cloud. If the username and password is valid, then the access is granted else the consumer is not allowed to enter into the system. After entering the cloud, the consumer can search a particular JAR file by providing certain keywords. If the file is found, the consumer can download the file.

4.1.4 Auditing Module

The auditing is carried out by a trusted Third Party Auditor (TPA).The data integrity, availability, security, reliability; authentication, confidentiality etc are checked by an audit system. Auditing is performed by a third party auditor. The audit records are provided whenever it is needed. The audit record contains the details of file usage such as the IP address of the system, Date, Time and duration of use. Push mode and pull

mode are the two distinct modes of auditing. Logs are periodically sent to the service owner in the case of push mode. But in the case of the pull mode, the service owner can retrieve the logs based on his needs.

4.1.5 Automatic JAR Checking Module

Consider a scenario in cloud computing as when there is a request from consumer for a JAR file, cloud provider provides the requested JAR file, but it may not belong to the actual service owner. That means, it may not be the JAR file which was provided or developed by the service provider.

That is, when a service owner wants to do business with his application after entering into a service level agreement or contract, he needs to upload his application on the cloud provider. It may become popular among the consumers of the cloud provider. In that situation, in order to gain extra revenue, the cloud provider may develop another application that provides the same service. And the cloud provider may provide his own JAR file for satisfying the consumer request. This JAR file provides the same facilities as that of the actual service owner's application. Service owner needs to detect this problem since it affects his business.

The technique to detect this problem is to simulate the consumer by the service owner. This is done by cross checking the checksum of the JAR file. If they do not match, service owner can conclude that his JAR file was not provided to the consumer upon request for the service. This simulation can be done at the provider side independently. So service owner can automatically check the trustworthiness of the cloud provider.

4.1.6 Integrity Checking Module

This is done to verify the integrity of the JAR at the consumer end. Whenever an application is uploaded in the cloud, the service owner automatically generates a checksum and it will be sent to the auditor while sending the JAR to cloud provider and auditor stores the same in its database. If the consumer tries to modify the data or application, the checksum may be altered and thus the auditor can detect misuses. This is carried out at the auditor end. The auditor verifies the automatically generated checksum with the checksum associated with the checksum associated with the downloaded JAR file. If any mismatches occur, it indicates the misuse of the downloaded file. Thus the auditor can deny the access to the JAR file.

4.2 Overview of the Framework

The service owner first authenticates itself to the CSP by providing username and password.If he is already subscribed to that CSP,then access is granted. After entering the system he creates executatble JAR file containing encrypted data. The JAR file is sent to the cloud.The authorised consumers are allowed to access the JAR files. The consumer can search for a particular file using a specific keyword. The consumer can download the files from the cloud. The auditor performs the auditing of the files. This audit record consists of the date, start time, end time etc.

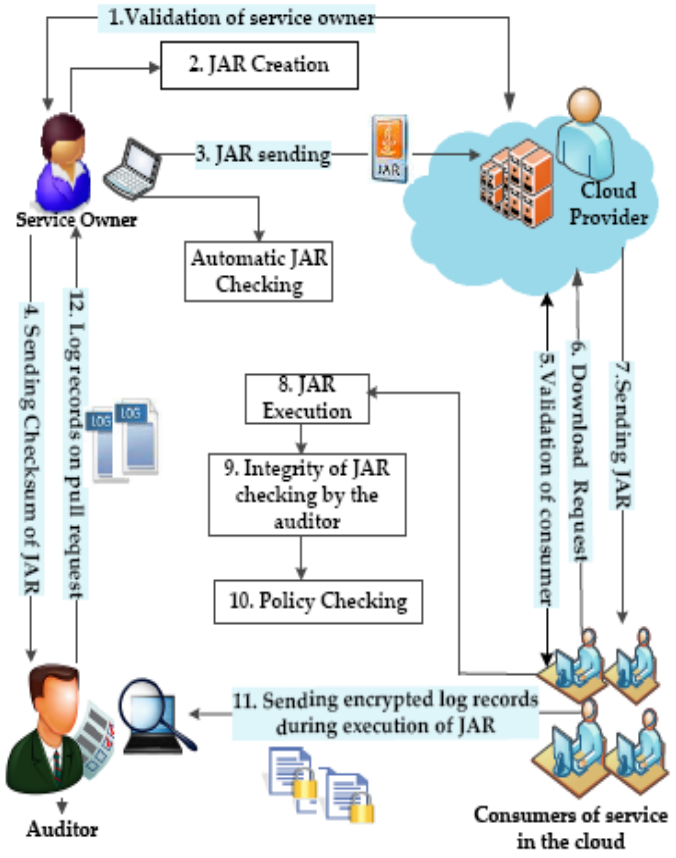

Fig 3 Overview of the framework assuring decentralized acountability in the cloud

4.3 Algorithms

4.3.1 Data Encryption Algorithm

Service owner needs to provide security to the data enclosed in the JAR file. RSA, DES etc are not suitable for digital image. In this paper, we have used a new secure algorithm for image encryption, which is based on RC4 streaming cipher algorithm and chaotic logistics map [4]. The algorithm works as follows. First, the external key is converted to initial value. Chaotic logistic map function [4] is used to generate a key stream using this initial value as input. A permutation is processed and the result is then XOR-end with the bytes stream of digital image. This encryption algorithm has the capability to develop the cipher-image cannot be easily understood using mere vision , cipher image and the plain image are free from statistical correlation, it will be very sensitive to key changes and it will not have change in contents of image during decryption and encryption process[4].

Chaotic Logistic Map Function

Chaotic logistic map [4] is a chaotic system which is popular. A CLM function [4] is givenin (1).

$$X_{n+1} = \lambda X_n (1 - X_n) \quad (1)$$

Where λ is a control parameter on the interval λ**= [0, 4]** and X_n is real number on the interval X_n**= [0, 1]** .This system is said to be chaotic if λ has a value on the interval λ**= [3.569955672, 4]**. In this paper, we use λ **=4** so the complete formula is shown in (2).

$$X_{n+1} = 4 X_n (1 - X_n) \quad (2)$$

Image Encryption Algorithm

The structure of encryption method consists of three main units. They are converter unit, CLM function unit, RC4 stream cipher unit. Key is converted to initial value X0by the converter unit**.** CLM function unit generates 256-bytes of array **V[i]** or also known as key array using the initial value X_0 **(**output of converter unit**)**.

The final step is RC4 stream cipher process where the content of the array **B[i]** (contents equal to 0 through 255 in ascending order) and array **V[i]** is interchangedwith each other and then the final result will be XORed with byte streams of cipher image to produce plain image or XORed with the plain image to produce cipher image[4].

```
Algorithm 1. Initial Permutation of array B
  1. j←0 to 255 do {
  2.  j←(j+B[i]+V[i])mod256
  3.  SWAP(B[i],B[j]) }
```

Fig 4 Algorithm for initial permutation of array B

```
Algorithm 2. Permutation Process(Permutate(i,j))
  1. i←(i+1)mod256
  2. j←(j+B[i])mod256
  3. SWAP(B[i],B[j])
  4. t←(B[i]+B[j])mod256
  5. return(t)
```

Fig 5 Permutation process

```
Algorithm 3 . Encryption Process for RGB channel
 1. i , j ← 0
 2. foreach(imageWidth){
 3.  foreach(imageHeight){
 4.  pixel←getPixel(imageWidth,imageHeight)
 5.  Permutate(i,j)
 6.  RED←(B[t] ⊕ pixel.RED)
 7. Permutate(i,j)
 8.  GREEN←(B[t] ⊕ pixel.GREEN)
 9. Permutate(i,j)
10. BLUE←(B[t] ⊕ pixel.BLUE)
11. /*Save Pixel Info*/
12.  image.SavePixel←(imageWidth,imageHeight,
     color(RED ,GREEN,BLUE)
13. } }
```

Fig 6 Encryption/Decryption process for each RGB channel

Initialization of array is done by storing values from 0 to 255. It is the first step in the algorithm for image encryption.

4.3.2 Algorithms for Logging and Auditing

Input: threshold value: dumping time specified by service owner, pull: command from service owner for log record.

```
1. while ( true) {
2. if (time < threshold value || pull ==0) {
3. Proceed to encrypt the log record ;
4. /* Start communication with the auditor,
5. If no response ,log record indicates an error*/
6. }
7. else //time>=threshold value || pull !=0
8. {
            JAR simply dump/push the log record
9. } }
```

Fig 7 Log retrieval algorithm

```
1. // Check permission before giving access to the data
2. // Send ipaddress of the consumer to cloud provider
   and will return the access permission of that consumer
3. if(permission = = 0)
4. // Privilege to download and view not present
5. else if (permission = = 1)
6. // Grant only view privilege
7. else
8. // Give download privilege
```

Fig 8 Permission checking algorithm

5. EXPERIMENTAL RESULTS

5.1 Experimental Settings

The experiment was done on a system with an Intel(R) Core(TM) i3-2310M processor running at 2.10 GHz, 4 GB of RAM. Algorithms are written and implemented in java platform and edited using Net beans editor. Results presented here are values from 9 continuous trials .The factors chosen for analysis

are size of JAR files and time taken for creation of JAR; log files mainly in milliseconds File size against computational cost and security strength per computational cost were also measured.

5.2 Analysis of Performance

In the experiments, the difference between size of the jar files and original files is analyzed. The size of JAR file is smaller than the original data since JAR format allows compression of files for efficient storage. Creation of log file is analyzed with respect to time taken by the system. As the size of original data increases, the size of JAR file also increases. It was found that the security strength increases with encryption and checksum calculation. Computational cost increases with the file size.

5.3 Time for JAR Creation

The time for JAR creation is recorded by analyzing the time for listing the files in specified directory and archiving the data and class files into a single JAR file. The time for creating JAR increases with increase in data size.

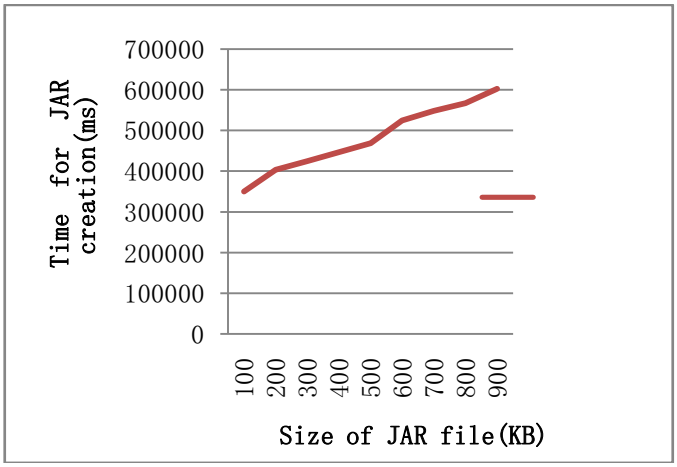

Fig 9 File size against time for JAR creation

The time for JAR creation is less when compared to the existing system. In the existing system, java policy file is added to the JAR .But in proposed system ,policy checking is not done using the java policy file.

5.4 Computational Cost Analysis

The computational cost of existing and proposed system is compared and it is plotted in the graph shown in figure 8.Computational cost increases with file size. A file of size 10KB has computational cost (with checksum calculation) of .3 ms and a file of size 60KB has computational cost of .8ms in the proposed system. But in existing system, the computational cost for 10KB of file is .1 ms. So from the graph shown infigure 8,we can understand that the computational cost of proposed system is slightly greater than that of existing system. It is due to the addition of class files for integrity checking, data encryption and policy checking. In the proposed system, the checksum calculation to verify the integrity of the JAR and the communications between the auditor etc added the computational cost than the existing system .In the existing system, the computational cost is .1 ms for a JAR file of 10KB.But with checksum calculation, the computational cost is raised to .3 ms. Even though the computational cost increased due to checksum calculation, security of the JAR file is enhanced.

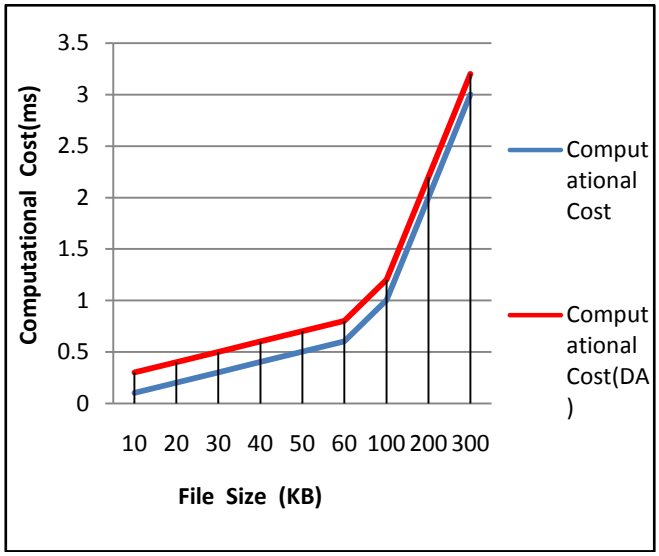

Fig 10 File size against computational cost

In the figure 8, computational cost(DA) stands for propose system and the other one corresponds to existing system.

5.5 Security Analysis

Security strength increased due to the addition of encryption and checksum calculation. Due to the strength of encryption technique, the security of the data provided inside the JAR file is very high. Security strength is measured in ms. In the graph shown in figure 9 ,the security strength per computational cost of existing and proposed sytem is compared. The security strength of proposed system is high due to data encryption technique, checksum calculation etc. In the graph shown in figure 9, security strength per computation cost (DA) corresponds to the proposed system and the other one belongs to the existing system. So it is clear that the security strength has been increased due to the additional layer of security that we have given. In the existing system, the policy of the user is checked using java policy file. And the encryption algorithm used in this paper[4] is capable to develop the cipherimage cannot be understood by mere vision and the cipher image and the plain image are free from the statistical correlation, it will be very sensitive to key changes and will not have change in contents of the image during decryption and encryption process.

Because of all these features, the security of the proposed system is higher than that of the existing system.

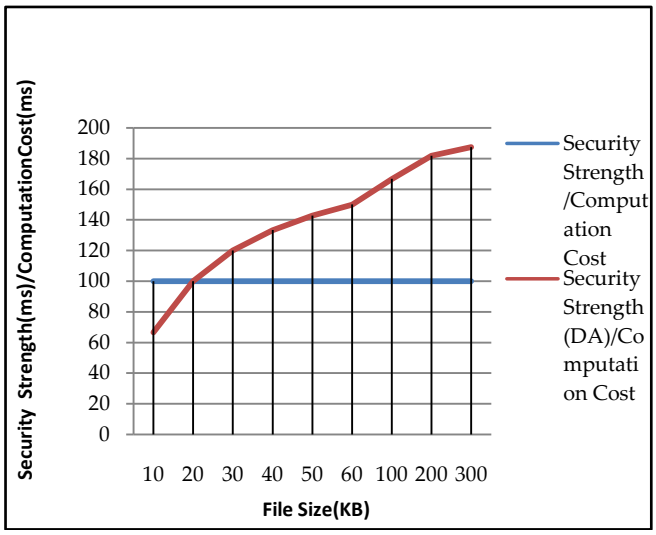

Fig 11 File size against security strength/computation cost

In the figure 11, the file size is plotted against security strength per computational cost.

5.6 Access Privileges

A newly registered user can either upload the JAR or can search for a JAR depending on the user type. User can access the JAR only based on their access privileges. A user can only view the data if he does not have downloaded right. In the existing system, service owner cannot ensure that the cloud provider is providing his application for the customer request. But in the proposed system, the service owner can automatically check whether this problem is there with the cloud provider by using a simulation process.

6. CONCLUSIONS

The proposed framework aims at automatically logging any access to the application of the service owner in the cloud along with a mechanism to perform distributed auditing. This approach provides the service owner a complete control over the application even in cloud. The main features of this work are that it helps the service owner to audit even copies of data that are created without his concern. The automatic JAR checking mechanism, policy checking and integrity checking of JAR are the important features of this framework. In the proposed work, in order to reduce overhead in the service owner, a third party auditor is used.

In the future, this work can be extended to provide additional layer of security to the application of the service owner. This work can be enhanced to support mobile applications also.

ACKNOWLEDGMENTS

This work was supported by the Department of Computer Science & Engineering, TKMIT,Kollam.We wish to thank all the faculty members of the department for their suggestions to improve this work.

REFERENCES

[1] Daniele Catteddu1 et al.,"Towards a Model of Accountability for Cloud Computing Services", Cloud Security Alliance, Hewlett-Packard,Tilburg University,2011.

[2] M.C. Mont, S. Pearson, and P. Bramhall, "Towards Accountable Management of Identity and Privacy: Sticky Policies and Enforceable Tracing Services, " Proc. Int'l Workshop Database And Expert Systems Applications (DEXA), pp. 377-382, 2003.

[3] S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud",Proc. First Int'l Conf. Cloud Computing, 2009.

[4] Ginting, R.U , Dillak, R. ," Digital Color- Image Encryption Using RC4 Stream Cipher and Chaotic Logistic Map,"Proc. IEEE Int'l Conf. Information Technology and Electrical Engineering, 2013

[5] A. Squicciarini, S. Sundareswaran, and D. Lin, "Preventing Information Leakage from Indexing in the Cloud", Proc. IEEE Int'l Conf. Cloud Computing, 2010.

[6] Ryan K L Ko1, Bu Sung Lee, Siani Pearson ,"Towards Achieving Accountability, Auditability and Trust in Cloud Computing", Cloud and Security Lab, HP Labs, Fusionopolis, Singapore;2009

[7] Y. Chen et al., "Oblivious Hashing : A Stealthy Software Integrity Verification Primitive", Proc. Int'l Workshop Information Hiding, F. Petitcolas, ed., pp. 400-404;2003.

[8] Anna. C. Squicciarini, Smith Sundareswaran, Dan Lin,"Ensuring Distributed Accountability for Data Sharing in the Cloud", IEEE Transactions Dependable And Secure Computing, Vol. 9, No.4; July/August 2012.

[9] B. Chun and A.C. Bevier, "Decentralized Trust Management and Accountability in Federated Systems," Proc. Ann. Hawaii Int'l Conf.System Sciences (HICSS); 2004.

[10] S. Pearson, Y. Sheen, and M. Mow bray, "A Privacy Manager for Cloud Computing," Proc. Int'l Conf. Cloud Computing (CloudCom);pp. 90-106, 2009.

[11] P.T.Jaeger, Jilin, and J.M. Grimes, "Cloud Computing and Information Policy: Computing in a Policy Cloud?",Journal of Information Technology and Politics, vol. 5, no. 3, pp. 269-28; 2009.

[12] A.Pretschner, M.Hilty, F.Schuo¨ to, C.Schaefer, and Thwarter, "Usage Control Enforcement: Present and Future," IEEE Security & Privacy, vol. 6, no. 4, pp. 44-53; July/Aug. 2008

[13] Abdul Wahid Khan et al ,"A Literature survey on data privacy issues and challenges in cloud computing",Proc of Into Journal ,IOSR,vol 1,issue 3;2012

[14] J.W. Helford, W.J. Catelli, and A.W. Rhodes, 'Using Self- Defending Objects to Develop Security Aware Applications in Java',Proc. 27th Australasian Conf. Computer Science, vol. 2;,pp. 341-349, 2004
[15] Cong Wang and Chui Ran ,"Toward Publicly Auditable Secure Cloud Data Storage Services",Proc of IEEE Network; 2010
[16] Kevin W Hamlen et al ,"Policy Enforcement Framework for Cloud Data Management",IEEE Computer Society Technical Committee on Data Engineering,2012
[17] Gowtham Gajala ,"Cloud Computing : A State of Art of The Cloud" ,Proc.Int'I Journal of Computer Trends and Technology,volume 4,Issue 1,2013
[18] Nishana Rahim ,"Secured Image Sharing and Deletion in the Cloud Storage Using Access Policies",Proc of Int Journal,IJCSE,vol 5,no 4,2013
[19] Swathi Sambangi," Cloud Data Storage Services Considering Public Audit for Security",Proc of Global Journal of Computer Science and Technology Cloud and Distributed,vol 13,Issue 1,2013
[20] Deveeshree Nayak ,"Empowering Cloud Security Through SLA",Proc of Int Journal ,JGRCS,vol 4,no 1,2013
[21] Ravi Shankar ,"A Comprehensive Security Model for Image Storage in Cloud',Proc Int Journal ,IJCTT,vol 4,Issue 3,2013
[22] Divya Rastogi ,"Study of Security Issues In Various Layers in Cloud Computing",Proc of Int Journal ,IJRREST,vol 2,Issue 3,2013
[23] Prabodh s Nimat ,"Efficient Data Sharing In Cloud With Third Party Auditor:A Review Study",Proc of Int Journal ,JEC&AS,vol 2,no.6,2013
[24] D. Boneh and M.K. Franklin, "Identity-Based Encryption from the Weil Pairing," Proc. Int'l Cryptology Conf. Advances in Cryptology, pp. 213—229; 2001.
[25] J.H. Lin, R.L. Geiger, R.R. Smith, A.W. Chan, and S. Wanchoo, "Method for Authenticating a Java Archive (jar) for Portable Devices",US Patent 6,766;353, July 2004.
[26] Ms Heena and Mr.Deepak ,'Building Trust In Cloud Using Public Key Infrastructure",Proc.of Int Journal ,IJACSA,vol 3,No;3,2012
[27] Son am and Satheesh ,"Access Control Based Data Security In Cloud Computing",Proc of Int Journal ,IJERA,vol2,issue 3,2012
[28] Andrew W Apple and Edward W Felten,"Proof Carrying Authentication'"Proc of ACM conference on Computer Communications Security;1999
[29] 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.
[30] Vaishali Singh ,"Revisiting Cloud Security Issues And Challenges",Proc of Int Journal ,IJARCSSE,vol 3,issue 7,2013