Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 8, August 2014, pg.497 – 506

RESEARCH ARTICLE

ASIDS - Authenticated Secure and Integral Data Storage in Cloud

Anjali A[1], Joe Mathew Jacob[2]

M.Tech, CSE Dept., VJCET, India
[1] anjali.appu@gmail.com; [2] joemathewjacob@gmail.com

Abstract— The introduction of cloud computing has triggered a change in computer community. The varying requirements of users are faultlessly met by the cloud. The cloud is used for a variety of purposes including storage services. Cloud storage services allow users to store their data and thus reduce space consumption. But security and privacy of data stored in cloud is a major issue faced by the user's and service providers alike. This paper deals with the introduction of a secure system to provide data security and user security. The paper outlines an idea to provide a three-fold technique for ensuring security of data. The technique progresses in three independent phases. These are security through encryption, authentication through one time passwords and integrity checking using the idea of bilinear pairing.

Keywords: Auditing, Authentication Bilinear pairing, Cloud Computing, Data Storage, Encryption, Integrity, One Time Passwords, Security

I. INTRODUCTION

Cloud computing is an idea introduced to refer to 'distributed computing' over Internet (or any real time network). The term cloud technically is used to describe an agglomeration of objects. The term came from the idea that this collection of objects could visually appear from a distance as a cloud. Cloud computing is also the agglomeration of a variety of technologies and services that can be made available on demand. NIST [1] defined Cloud Computing as "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources like servers, storage, applications, and services , that can be rapidly provisioned & released with minimal management effort or service provider interaction". Increase in usage of Cloud and the various services provided by cloud has lead to some issues of concern that include reliability, availability of services and data, security, complexity, costs, regulations and legal issues, performance, migration, reversion, the lack of standards, limited customization and issues of privacy.

The evolution of digitized data had a great impact on our society. This was in the case that the society has become dependent on digital services. These services has a variety of functionality like extracting critical information and allowing decision making by different kind of peoples in all aspects of life. Thus, the introduction of cloud services and cloud-based infrastructures for storage began to be thought of as the next generation solution that offers reliance as size of data increases. In nutshell storage services refer to the variety of services provided by a cloud server and allow the users to store critical data like email, account details, and disk backups. These services became popular instantaneously and thus had an immediate and notable effect. Users began to store large volumes of data in cloud, or in cloud service providers. These CSPs (Cloud Service Providers) were usually companies with big-end servers and systems like Amazon, Google, Yahoo etc.

Originally the CSPs started renting out their storage space to customers. But even such big-end servers have limited (though very large) amount of memory. As the number of customers increase this memory is very likely to get depleted. With this inference came the issue of data integrity in the cloud.

A CSP that provides storage services may make a guarantee about the level of integrity it can provide. Failing behind in such standards may affect the reputation of the CSP and thus it is assumed throughout the paper that a CSP may resort to any means (legal or not) to ensure its users that their data is securely stored. But in reality that data may have been modified or completely or partially deleted. Such concerns lead to the need for privacy preserving and integrity checking protocols and mechanisms in Cloud.

The rest of the paper is organized as follows. In section II, the idea of One Time Passwords and their help in authentication of users is described. Section III describes the idea of encryption of file using AES Algorithm. Section IV describes the use of bilinear pairing for checking the integrity of any file uploaded by a user to the Server. Such checking is proven to be good for Cloud environments since the computation cannot be forged or duplicated at the Server or Auditor.

II. AUTHENTICATION

User authentication is an important part of any user oriented system. In any cloud environment it is essential for users to be authenticated to provide security for the uploaded data.

In the proposed security model one time password has been used for authenticating the user. The password is used to keep the user account secure and secret from the unauthorized user. But the user defined password can be compromised. To overcome this difficulty one time password is used in the proposed security model.

The overall approach is given in Fig. 1.

Passwords can be thought of as one of the most important parts for security and manageability of any system or service. However remembering passwords can be difficult for a User. This leads users to often create simple passwords or write them down to make sure that they'll remember them. In addition, there are small secure and inefficient procedures for resetting passwords.

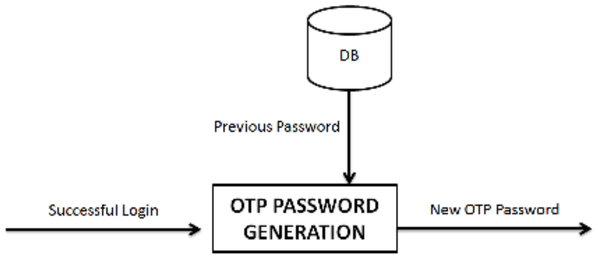

Fig. 1. Authentication Using OTP

While keeping in mind such limitations, mitigating these kinds of security related problems faced especially when remote users try to access your network is a difficult task. The password maintenance solution must be more robust.

There are a few methods that are already found that deals with eliminating standard passwords for your remote users. One of this is when one can use certificate authorities to issue certificates to your users, but this requires a public key infrastructure (PKI) [2]. The main disadvantage in using a PKI is that it is expensive to set up and maintain. It can also be a difficult task to maintain certificates for remote users, especially when a hardware-based token is used like smart cards. However such trade-off of high cost for high security is a common approach.

Another option is to use a standards-based OTP solution. Passwords are basically of 2 types –static and dynamic.

A static password is usually only changed when user manually wants to change it, or when it has expired a certain pre-specified time, or when the user has forgotten it and needs to reset it. Passwords are cached on computer hard drives and later stored on servers. Due to this reason, they are easy to crack. This becomes a serious concern in case of laptops since they can be easily stolen. But a simple static password solution can become more of a liability especially when remote users come to picture.

Unlike a static password, the one-time password, which are dynamic passwords, changes each time when user logs in. A popular example is the hash-based OTPs which uses cryptographic hashing algorithms to compute the OTP [3]. A cryptographic hash is a one-way function. It maps an input message of arbitrary length to a fixed-length digest. Thus, a hash-based OTP starts with the inputs, pass it through a one-way function, and produces the fixed-length output called OTP.

In our approach, the method for generating an OTP is made more secure through 2 ways. One is that, the password is computed using the previous OTP that is known only to the Server. Second is that the password is sent to the Client after the first step of login process.

In the first step, the Client logs in with his username. Just after this, a mail is send to the Client's mail id and this mail contains the required OTP. Using this retrieved OTP only, the Client can finish the login process.

The scheme is shown diagrammatically as in Fig. 2.

Thus whenever a user login in the system, he/she will be provided with a new password for using it in the next login. This is usually provided by the system itself. This password will be generated randomly. Each time a new password is created for a user, the previous password for that user will be erased from the system. New password will be updated for that particular user. A single password will be used for login only once.

By this system, existence of unauthorized user will be pointed out. The newly generated password is restored in the system after MD5 hashing. The main purpose of MD5 hashing is that this method is a one way system and unbreakable.

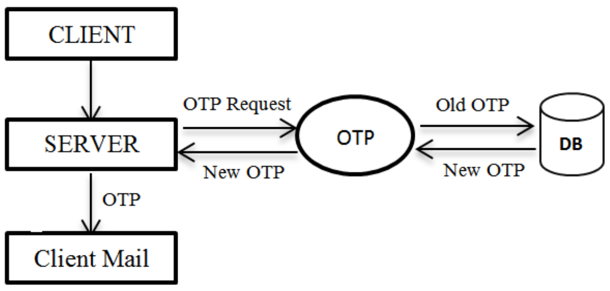

Fig. 2. OTP Generation

Therefore it will be difficult for an unauthorized or unknown party for retrieving the password for a selected user even if gained access to the system database.

III. File Encryption

A. Encryption

Data encryption techniques have become widely used to insure the protection of critical data or information, like credit card numbers. However whatsoever may the improvements be, privacy may still be compromised.

Encryption is a procedure that makes the contents of a file or message or any text input unintelligible to anyone other than those who are authorized to read it [4]. Here the input data undergoes a translation resulting into a secret code. This is considered to be one of the most effective ways to achieve data security. To read an encrypted file the authorized person must have access to a password that enables to decrypt it. Encrypted data is referred to as a cipher text. Decryption can be considered as the process of converting an encrypted data back to its original form with the help of a password.

Encryption technology has improved rapidly over time, and it is now used practically by even individual peoples. There is a continuous developing trend and introducing easy and new ways to ensure privacy online. However, no data can be thought of as one hundred percent safe.

B. Need For Encryption

Data security includes the following 4 basic functions: [5]

• Confidentiality: protection against data leakage to third parties.
• Integrity: protection against alteration of prepared data.
• Authenticity: that guarantees authorized usage.
• Accountability : that checks for errors periodically.

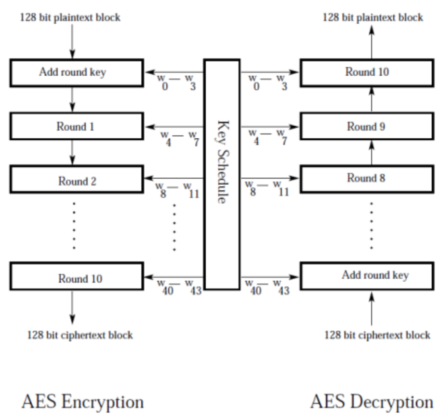

Fig. 3 Overall AES Workflow

C. AES Encryption

The Advanced Encryption Standard (AES) [6] is an encryption algorithm for securing sensitive information that has gained tremendous popularity in the present age.

AES became popular due to the fact that it offers security to protect data for the next 20 to 30 years. It is easy to implement and offer good defenses against various attacks. The workflow of an AES Algorithm is shown in Fig. 3.

D. Working Of AES

1) Block

Before applying the algorithm to the data, the key sixe and block must be determined. AES allows for block sizes of 128, 168, 192, 224, and 256 bits. AES allows key sizes of 128, 192, and 256 bits. The standard encryption uses AES-128 where both the key size and blocks are 128 bits. The key size is commonly denoted as Nk and the block size is commonly denoted as Nb. Nb refers to the number of columns in the block where each row in the column consists of four cells of 8 bytes each for AES-128.

Each character is stored in a cell of the block. The blank cells shown in the diagram are not really blank as they represent the spaces in the text. Depending on how the algorithm is implemented the characters may be stored as integer values, hexadecimal values, or even binary strings. All three ways represent the same data. Most diagrams show the hexadecimal values, however integer and string manipulation is much easier to do when actually programming AES. The plain text is stored into blocks column by column and block by block until all the data is stored.

When the data is not a multiple of the block size some form of padding must be used. Padding is when extra bits are added to the original data. One forms of padding includes adding the same bytes until the desired size is reached. Another option is padding with all zeros and having the last byte represent the number of zeros. Padding with null characters or random characters are also forms of padding that can be used.

2) Key

Once a form of padding is chosen the data is represented as some number of complete blocks. The last thing needed before using the algorithm is the key. The key also known as the cipher key is also the same size as the block in this example.

Unlike most data and transformations the cipher key can have any values chosen by the designer with no restrictions as long as the key is the correct length. The key is also stored as a block similar to the plain text

The proposed scheme uses the following algorithm that is based on the property of bilinear pairing to generate the key required for AES Algorithm.

```
Algorithm 1 : AES Key Generation
Step 1. Password = Pin Number of User, provided during registration.
Step 2. Convert Password to its integer equivalent, Xpass :
a) Take integer equivalent of each character in string.
b) Sum the total
c) Divide, if necessary, by “limit” to get value in a specified range.
Step 3. Compute e( G1 x pass , G2) as the required key.
```

3) High Level Algorithm Overview

KeyExpansion

Round keys are derived from the cipher key in this step. AES uses a separate 128-bit round key block for each round and an additional one more.

b) InitialRound
AddRoundKey – Here each byte of the state is taken separately and is combined with a block of the round key using bitwise xor operation.

c) Rounds

SubBytes – This is a substitution step that is non-linear in nature. Here each byte is replaced with another according to information provided by a lookup table.
ShiftRows – This is a transposition step. Here the last three rows of the state are shifted cyclically a certain number of times.
MixColumns – This is basically a mixing operation which operates on the columns of the state. The step combines four bytes in each column.
AddRoundKey – Adding using XOR operation, the round key.

d) Final Round (no MixColumns)
SubBytes
ShiftRows
AddRoundKey.

5) Overview Of Steps

a) SubBytes
This step is called SubBytes. It is basically a byte-by-byte substitution. The corresponding substitution step used during decrypting data is called InvSubBytes. Here a 16 × 16 lookup table is used to replace any byte from the input state array.

b) ShiftRows
This step shifts the rows of the state array. The corresponding transformation during decryption is denoted InvShiftRows for Inverse Shift Row Transformation.

c) MixColumns
This step mixes up the bytes in each column during the forward process. The corresponding transformation during decryption process is denoted InvMixColumns and stands for Inverse Mix Column Transformation.

d) AddRoundKey
This is for adding the round key to the output of the previous step during the forward process. The step during decryption is called InvAddRoundKey for Inverse Add Round Key Transformation.

After connecting with the system a user can upload or download the file(s). For the first time when connected with the system the user can only upload file(s). After that users can both upload and download their files. When a file is uploaded by an user the system server encrypts the file using AES encryption algorithm. In the proposed security model 128 bit key is used for AES encryption. 192 bit or 256 bit can also be used for this purpose. The security of the key used is assured by the fact that it is computed each time and the key as such is not stored.

IV. INTEGRITY CHECKING

Here an auditing system for cloud storage is proposed and as shown in Fig. 4 which involves the cloud server (server), data owners (owner), and the third-party auditor (auditor). The owners create the data and store their data in the cloud. The cloud server stores the owners' data securely and provides the data access to users. These users are also called as data consumers. The auditor is a trusted third-party that has expertise and capabilities to provide data storage auditing service for both the owners and servers.

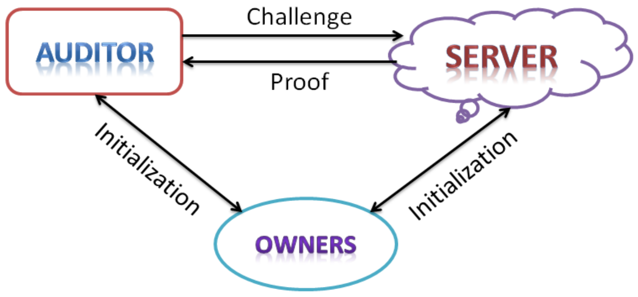

Fig. 4 System Model

The section first describes the basic idea of bilinear pairing before explaining the algorithms at the client, server and auditor side.

A. Bilinear Pairing

Pairings in general can be used to transport the discrete logarithm problem on a certain class of elliptic curves over a finite field to the discrete logarithm problem on a smaller finite field, where a sub-exponential index calculus attack can be used to attack the problem [8]. However, it was the publication of an identity based encryption scheme (by Boneh and Franklin) [9] based on bilinear pairings that triggered a real upsurge in the popularity of pairings among cryptographers. Following Boneh and Franklin, a lot of cryptosystems based on pairings have been proposed which would be hard to construct using more conventional cryptographic primitives.

A pairing is map $e : G1 \times G2 \rightarrow GT$ where G1, G2 and GT are groups (usually cyclic of the same prime order). In some instances the pairing is symmetric i.e G1 = G2. If not, the pairing is said to be asymmetric.

Bilinear Pairing is a Pairing that satisfies property of bilinearity, which means the following should hold:

$$e(aP, bR) = e(P, R)^{ab} \qquad (1)$$

P in G1, Q in G2 and all a, b in Z.

A pairing is admissible if the mapping is also non-degenerate and computable. Non-degeneracy means the mapping cannot be the trivial map which sends every pair of elements of G1 and G2 to the identity element of GT. Because all are groups of prime order, it follows that if P is a generator of G1 and Q is a generator of G2, then e(P, Q) is a generator of GT. A mapping is said to be computable if an algorithm exists which can efficiently compute e(P, Q) for any P, Q. If G1 = G2 then the pairing is said to be symmetric. Otherwise it is said to be asymmetric.

B. Use Of Bilinear Pairing

The following subsections illustrate the algorithms used to implement bilinear property to ensure integrity. The need for integrity is widely discussed in [10]. The basic framework for integrity checking is derived from the idea proposed in [7]. The algorithms are explained in subsequent sections.

1) Algorithm : Client side

At the Client side, on receiving a file for uploading to the server, the following processing is done to generate the metadata.

a) Division of file to blocks.

At first, the input file is divided into blocks. This can be done in 2 ways. The file can either be divided into a pre-specified number of blocks or the file can be divided into blocks of a pre-specified size. This approach divides the file into a pre-specified number of blocks, each of the same size. This is as shown in Fig. 5.

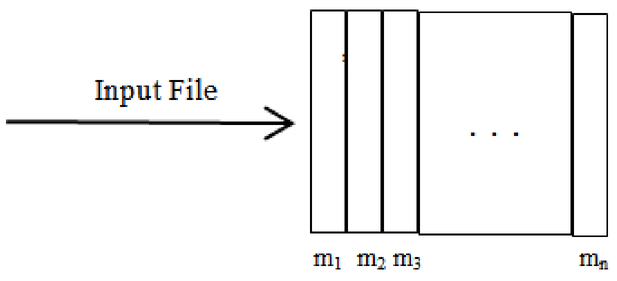

Fig. 5 Division Of Files To Blocks

b) Defining the Pairing Function.
A Pairing "e" is defined over 2 groups G1 and G2 over a field gt.

c) KeyGen
This key generation algorithm takes two random parameters as the secret tag key 'skt' and secret hash key 'skh'. It calculates a public tag key pkt as

$$pkt = g2\ skt \qquad (2)$$

where g2 is the generator of G2.

d) TagGen
The tag generation algorithm takes inputs as an encrypted file M, the secret hash key skh, and the secret tag key skt. For each data block mi, it computes a data tag ti based on skt and skh. It outputs a set of data tags T.
A data tag ti is calculated for each block as

$$ti = g1\ (hi.mi) \qquad (3)$$

Where hi is the hash value for a block that is calculated using skh, mi is the unique value for a block.

After this the client stores metadata associated with the file, at both auditor and server.
At the auditor it saves,
- Tag value of each block.
- Hash value of each block
- Public Tag Key, Pkt
- Total number of blocks.

At the Server, the Client saves :
- Tag value of each block.
- Hash value of each block
- Total number of blocks

2) Algorithm : Auditor side

The request for a file verification is send by the client to an auditor. After verification, the auditor sends back the response. The complete flowchart is shown in Fig. 6.

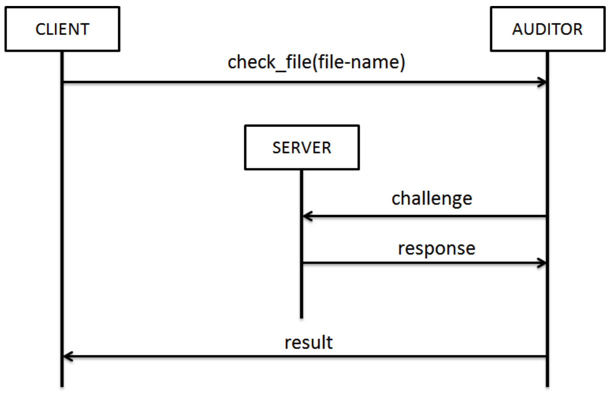

Fig. 6. File Verification

There are 2 phases for the auditing protocol to implement auditing functionality at the auditor.

The first is the challenge generation phase. This phase is initiated when a client wants to verify a particular file at the server. This request is send to the auditor which generates a challenge based on the metadata of the file and some random values.

The second phase is the verification phase which is initiated after the Server responds to the challenge from the Auditor. This phase checks whether the response from the Server is correct by using the property of bilinear pairing.

3) Challenge Phase

The challenge algorithm takes as input the abstract information of the data Minfo (e.g., file identity, total number of blocks, version number, time stamp, etc.). It outputs a challenge C by the following steps:

- Take 3 random numbers x, vi, r
- Calculate U as g1x and R as pkt r
- Generate challenge C for each challenged block with U , vi, and R.

4) Verification phase

The server responds back with 2 values :

- Data Proof DP
- Tag Proof TP

The verification is done according to the following steps :

- Calculate Hchall as pkt h.vi
- Calculate hc = e(U, Hchall)
- Calculate tc = e(u.vi, pkt)
- Calculate eTP = e(TP, pkt x.r)
- Calculate LHS = DP x hc
- Calculate RHS = eTP x tc

If LHS = RHS, the file is not modified. If not, the file is modified.

5) Algorithm : Server side

The server outputs a a proof P.
The proof consists of the data proof DP and the tag proof TP.
The tag proof is generated as :

TP= ti x vi (4)

To generate the data proof,

- Calculate MP as

MP= vi x mi x hi (5)

- Generate the data proof DP as

$DP=e(U,R)^{MP}$ (6)

These 2 values are sent to the Auditor for the verification phase.

SIMULATION AND EVALUATION

A. Simulation Using CloudSim

It is a holistic software framework for modeling Cloud computing environments and performance testing application services.CloudSim offers the following novel features:

• Support for modeling and simulation of large scale Cloud computing infrastructure, including data centers on a single physical computing node.
• A self-contained platform for modeling service brokers, data centers, provisioning, and allocations policies.
• Support for simulation of network connections among the simulation elements.
• Facility for simulation of federated data center environment that inter-networks resources from both private and public domains.

The simulation workflow is shown as in Fig. 7.

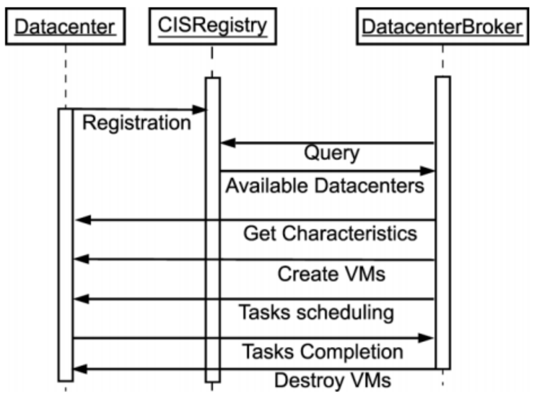

Fig. 7. Simulation WorkFlow

B. Evaluation

The system is evaluated in 2 ways.

- Block Processing Time
- Auditing Time Comparison

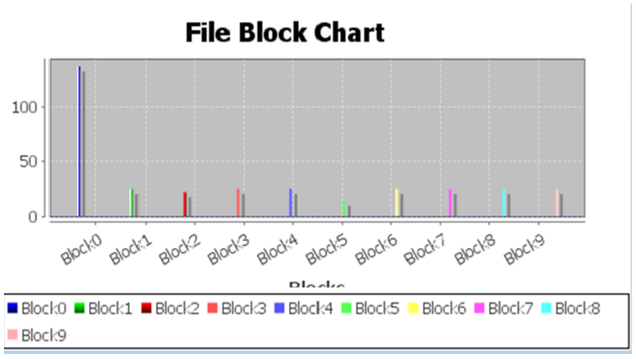

Fig.8. Block Comparison Chart

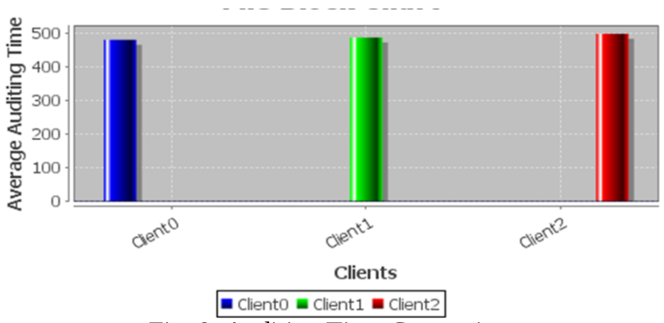

Fig. 9. Auditing Time Comparison

The block processing time gives the processing time for division of file into blocks and calculation of metadata for each block. Auditing time comparison gives the time required for auditing (challenge generation and verification) by different clients.

This is shown in Fig. 8. and Fig. 9.

CONCLUSION

Data storage security in Cloud is an important issue. The security includes authentication of users, encryption of data, and integrity checking of data stored in cloud. The security for data stored in cloud can be achieved through encryption. In any Cloud based system, support for authentication of users must be provided that further improve security. By comparing the various approaches for data security, especially data integrity, an integrated approach that includes these 3 areas is proposed and implemented. The property of bilinearity pairing is used for ensuring integrity. Here, data owners store their data on cloud servers and users (data consumers) access the data from cloud servers. In an error-free system, the system must prove to be trustworthy to the data owners, so that once the data get stored in the Cloud, data owners can remove their copy of data without fear of losing their data. The scheme also considers dynamic updates to the data in cloud. The proposed scheme provides higher level of security through the combined features of encryption, authentication, and integrity checking while ensuring that the computation time required for processing is as small as possible.

ACKNOWLEDGEMENTS

The authors would like to thank the staff and students of Computer Science and Engineering Department, Mathematics Department and the management of Viswajyothi College of Engineering And Technology, Vazhakulam, India for their continued support and guidance.

REFERENCES

[1] Peter Mell, Timothy Grance "The NIST Definition Of Cloud Computing", September 2011.

[2] "Public Key Infrastructure", Explanation. url:http://www.entrust.com/what-is-pki/

[3] Bart PRENEEL "Analysis and Design of Cryptographic Hash Functions" , 2003.

[4] "Understanding Encryption", SANS Newsletter 'OUCH', July 2011.

[5] Karen Bennett, Patty Chiles, and Michele Jacobs , "An Educator's Guide to Privacy".

[6] Susan Landau, "Communications security for the twenty-first century: the advanced encryption standard."

[7] K.Yang and X.Jia "An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing" , IEEE transaction on parallel and distributed systems, 2013.

[8] Ran Canetti and Ron Rivest,"Pairing-Based Cryptography" , Special Topics in Cryptography, 2004.

[9] D.Boneh and M.Franklin, "Identity-Based Encryption from the Weil Pairing".

[10] Cong Wang and Kui Ren,Wenjing Lou, Jin Li, "Toward Publicly Auditable Secure Cloud Data Storage Services" , IEEE Network 2010.

[11] Data Storage Services In Cloud – Internet url:http:// computer.howstuffworks.com/cloud-computing/cloud-storage1.htm"

[12] Yanpei Chen, Vern Paxson, Randy H. Katz, "What's New About Cloud Computing Security?", Electrical Technical Report No. UCB/EECS-2010-5