Internet of Things and Cloud Computing
2018; 6(1): 25-35
http://www.sciencepublishinggroup.com/j/iotcc
doi: 10.11648/j.iotcc.20180601.14
ISSN: 2376-7715 (Print); ISSN: 2376-7731 (Online)

Secure Cloud Auditing Over Encrypted Data

Yesu Ragavi Kannadasan, Rajagopal Devarajan

Department of Master of Computer Applications, Vivekanandha College of Arts and Sciences for Women, Tiruchengode, India

Email address:
ragavikannanmca@gmail.com (Y. R. Kannadasan), sakthiraj2782007@gmail.com (R. Devarajan)

To cite this article:
Yesu Ragavi kannadasan, Rajagopal Devarajan. Secure Cloud Auditing Over Encrypted Data. *Internet of Things and Cloud Computing*. Vol. 6, No. 1, 2018, pp. 25-35. doi: 10.11648/j.iotcc.20180601.14

Received: February 8, 2018; **Accepted**: March 8, 2018; **Published**: April 13, 2018

Abstract: Cloud computing is a popular technology which permits storing and accessing data over Internet instead of storing it on local machines' hard drive. Cloud users can enable to store their data on cloud without any anxiety about its accuracy and reliability. However storing data on cloud imposes certain security challenges. Outsourcing data in cloud result may lose physical control over their data. Certain Cloud Service Providers (CSPs) may operate dishonestly with the cloud users' data, they may sneak the data from cloud and sell it to third parties in order to earn profit. Even though outsourcing data on cloud is inexpensive and reduces long duration storage and maintenance complexity, there is least assurance of data integrity, privacy, security and availability on cloud servers. A number of solutions have been recommended to solve the security issues in cloud. This paper mainly focuses on the integrity verification strategy for outsourced data. The proposed scheme combines the encrypting mechanism.

Keywords: Cloud Computing, Data Encryption, Security, Cloud Audit, Cloud Service Provider

1. Introduction

Cloud provides handling and maintaining large group of remote servers to be in a network, which is allowed the centralized data repository and access to the computer services or resources whenever required [1]. The user is concerned about the integrity of data stored in the cloud can be hacked or modified by outside attackers [2]. Therefore, a new concept called data auditing is introduced to check the integrity of data with the help of an entity called Third Party Auditor (TPA). In a cloud-computing environment, data and the application are controlled by the CSP. This leads to a usual concern about data protection from internal and external threats [3]. In cryptography, encryption is the process to encode the messages or informations, which will help the authorized parties, can read it. Encryption itself does not prevent interception, but denies the message content to the interceptor. For technical reasons, an encryption scheme usually uses a pseudo-random encryption key generated by an algorithm [4]. A third party auditor has some capabilities to provide more work that is efficient and convince both cloud service providers and owners. For the third party auditing in cloud storage systems, there are several important requirements [5]. The auditing protocol should also be able to support the batch auditing for multiple owners and multiple clouds. This paper work focuses on brief descriptions of various public key cryptography algorithms [6]. As users no longer physically possess the storage of their data, traditional cryptographic primitives for the purpose of data security protection cannot be directly adopted. Thus, how to efficiently verify the correctness of outsourced cloud data without the local copy of data files become a big challenge for data storage security in Cloud Computing [7].

2. Literature Survey

Sarah sheikh and Deepali vora et al. (2017) focus on their work integrity verification strategy for outsourced data. Their proposed scheme combines the encrypting mechanism along with integrity verification strategy. They were concluded from the work that of eliminates the overhead of performing auditing task from the client and also lessons the cloud users concern that their uploaded data may be accessed by an un-trusted organization or individual.

Kalyani sonaware and Rahil dagade et al. (2017) state that the cloud service providers (CSP's) can access users sensitive data without authorization. They made a wide survey on various searching techniques towards effective data utilization in cloud storage. The main goal of their work to prevent the cloud server

from document set, the index file, and the user queries.

Swapnali more and Sangita chaudhari et al. (2016) state that the TPA (THIRD PARTY AUDITOR) performs the main role of data integrity check. It can performs activities like generating hash value for encrypted blocks received from cloud server, concatenating them and generate signature on it. It later compares both the signatures to verify whether the data stored on cloud is tampered or not. It verifies the integrity of data on demand of the users. The cloud server is used only to save the encrypted blocks of data. This proposed auditing scheme is developed by AES algorithm for encryption in cloud environment.

Nitin Nagar and Ugrasen suman et al. (2017) proposed Reliable and Enhanced Third Party Auditing system in Cloud Server Data Storage which provides better scalability, flexibility, high performance, availability and less storage cost as compared to other physical storage of data. However, the security of stored data is the major concerned for organizations and individual user to adopt cloud based environment. Their proposed approach uses the functionality such as, public verifiability, metadata generation, data dynamics, storage access point, encryption and decryption of data through RSA algorithm and IP range in case of private cloud.

Mahesh kumar N. B. discuss et al. (2017) detail about Encrypted Big data Using AES Deduplication in Cloud Storage. In that he states that users can achieve an effective and economical approach for data sharing among group members in the cloud with the characters of low maintenance and little management cost. Meanwhile, it must provide security guarantees for the sharing data files since they are outsourced. Because of the frequent change of the membership, sharing data while providing privacy preserving are still challenging issues, especially for an untrusted cloud due to the collusion attack. which means previous users need not to update their private keys for the situation either a new user joins in the group or a user is revoked from the group. The proposed scheme provides a secure approach to protect and deduplicate the data stored in cloud by concealing plaintext from both CSP (Cloud Service Provider) and AP (Authorization Party). The security of the scheme is ensured by PRE theory, symmetric key encryption, AES and Elliptic curve Cryptography theory.

Sweta k. Parmer and k. c. Dave et al. (2013) implementation the concurrency model using verilog HDL, which is used to concurrent of process found in hardware elements. It is a gateway design. Verilog HDL is a general – purpose hardware description language that is easy to learn and easy to use. It is similar in syntax to the C programming language. Verilog HDL allows different levels of abstraction to be mixed in the same model. Verilog came into being as a proprietary language supported by a simulation environment that was the first to support mixed–level design representations comprising switches, gates, RTL, and higher levels of abstractions of digital circuits.

Pitchaiah. M, Philemon Daniel, and Praveen et al. (2012) discussed about implementation of advanced encryption standard algorithm. Cryptography is the study of mathematical techniques related to the aspects of information and security such as confidentiality, data integrity, entity authentication and data origin authentication. In data and telecommunications, cryptography is necessary when communicating over any unreliable medium, which includes any network particularly internet. Cipher and inverse Cipher are composed special number of rounds. For the AES algorithm, the number of rounds to be performed during the execution of the algorithm uses a round function that is composed of four different byte-oriented transformations. They are Sub Bytes, Shift Rows, Mix columns, and Add Round Key. Their implementation and results are different sub modules for AES algorithm by using Verilog code. This implementation will be useful in wireless security like military communication and mobile telephony where there is a gayer emphasis on the speed of communication. The encrypted cipher text and the decrypted text are analyzed and proved correct. The encryption efficiency of the proposed AES algorithm was satisfactory.

3. Existing Model

When directly applied in large collaborative data outsourcing in cloud environment, the existing model faced the following drawbacks. On the other hand, invariably sending back all files solely based on presence and absence of the keyboard future incurs large unnecessary network traffic, which is undesirable in pay-as-you-use cloud paradigm. The importance of data sharing and the need to ensure privacy and security is discussed in a number of existing articles.

Drawbacks of Existing Model

(i). Methodology for searchable encryption techniques, which allows to users without leaking information about the data itself & users request with security

(ii). Large unnecessary network traffic while transferring the data.

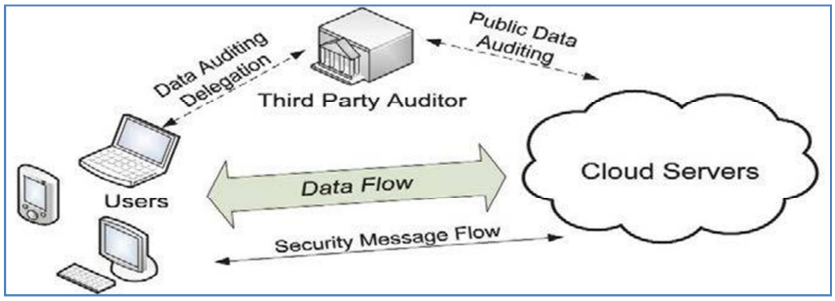


Figure 1. *Cloud Data Storage Architecture.*

4. Related Work

Cloud Audit is a specification for the presentation of information, which deals about how a cloud computing service provider addresses the control frameworks. The goal of Cloud Audit is to provide cloud service providers with a way to make their performance and security data readily available for potential customers.

Data auditing is the process of conducting a data audit to assess how company's data is fit for given purpose. This involves profiling the data and assessing the impact of poor quality data on the organization's performance and profits.

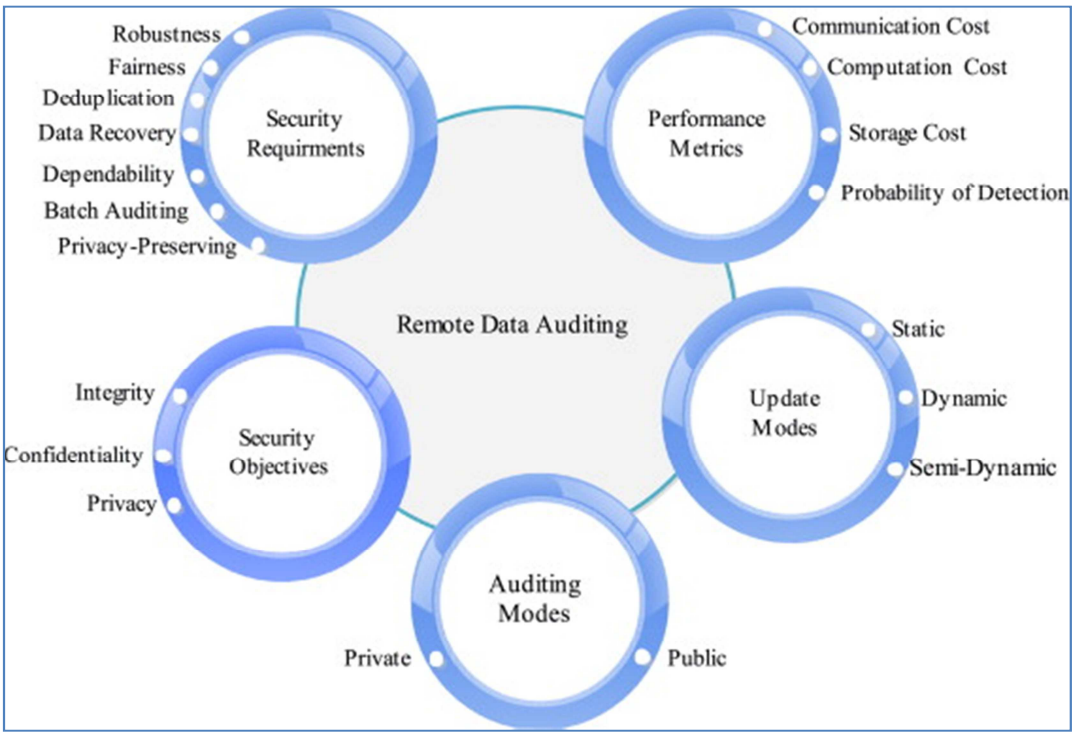


Figure 2. *Cloud Data Storage Architecture.*

TYPES OF DATA AUDITING

a) Compliance audit
b) Construction audit
c) Financial audit
d) Information systems audit
e) Investigative audit
f) Operational audit
g) Tax audit

4.1. Batch Auditing

Third-party auditor (TPA) is used to check the integrity of outsourced data. Privacy preserving public auditing mechanism is used to verify the data integrity with privacy. TPA supports auditing for multiple users simultaneously. Batch auditing mechanism is used for multi user environment

TYPES OF BATCH AUDITING

1. Public auditing
2. Private auditing

4.2. Third Party Auditor

A new concept called data auditing is introduced to check the integrity of data with the help of an entity called Third Party Auditor (TPA).

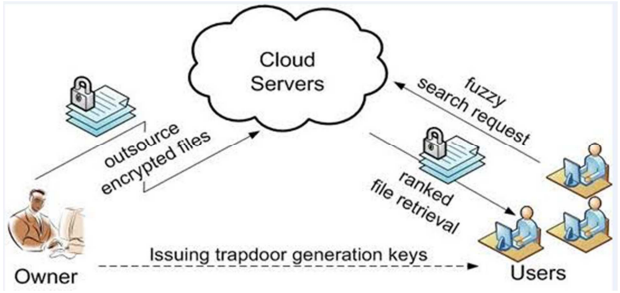


Figure 3. *Third party Auditor.*

It later compares both the signatures to verify whether the data stored on cloud is tampered or not. Why we need TPA? Third–party prequalification is a rapidly growing business model providing companies with a convenient way to ensure contractors meet safety, fiscal, and other requirements before being hired to work on a jobsite. How the Process Works? Companies, often called Owner Clients or Owner Operators, and contractors both pay to secure an account with a third-party auditor, such as ISN networld, PICS, PEC premier, BROWZ, Comply works, veriforce, texura and others.

4.3. Data Protection from External and Internal Threats

Virtual machines live their lives as disk images that are

hosted on a hypervisor platform and are easily copied or transferred to other locations. This mobility is advantageous because it allows VMs to be transported to other physical machines via an image file that defines the virtual disk for that IDENTIFYING CLOUD COMPUTING SECURITY RISKS 69. Unfortunately, the ability to move and copy VMs poses a security risk because the entire system, applications, and data can be stolen without physically stealing the machine "From a theft standpoint, VMs are easy to copy to a remote machine, or walk off with on a storage device".

4.3.1. Threats for Cloud Service Users

a) Loss of Governance
b) Loss of Trust
c) Unsecure Cloud Service User Access
d) Lack of Information/Asset Management

4.3.2. Threats for Cloud Service Providers

a) Evolutional Risks
b) Business Discontinuity
c) License Risks Software
d) Bad Integration
e) Unsecure Administration API
f) Shared Environment
g) Service Unavailability
h) Data Unreliability

4.4. The Process of Cryptography

Cryptography is closely related to the disciplines of cryptology and cryptanalysis. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as clear text) into cipher text (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers.

The ordinary information (called plaintext) is converted into unintelligible text (called cipher text). Decryption is the reverse, in other words, moving from the unintelligible cipher text back to plaintext.

4.4.1. Components

a) Confidentiality (the information cannot be understood by anyone for whom it was unintended)
b) Integrity (the information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected)
c) Non-repudiation (the creator/sender of the information cannot deny at a later stage his or her intentions in the creation or transmission of the information)
d) Authentication (the sender and receiver can confirm each other's identity and the origin/destination of the information)

4.4.2. Architecture

In computing the java cryptography architecture (JCA) is a framework for working with cryptography using the Java programming language. It forms the part of the Java security API, and was first introduced in JDK 1.1 in the java. security package. The JCA uses a "provider"-based architecture and contains a set of APIs for various purposes, such as encryption, key generation and management, secure random-number generation, certificate validation, etc. These APIs provide an easy way for developers to integrate security into application code

4.4.3. Algorithm Approach

a) Asymmetric cryptosystem of Paillier is applied for encryption of *l+1* images, where one is the secret image to be shared and all the other are individual secret images used for shared trust and security.
b) Due to additive homomorphic property of Paillier, addition operation over the plain text will give same result as multiplication over ciphered text.
c) Extraction of secret image is possible only if the individual secret images are available.

4.4.4. Differences Between Algorithms

Table 1. *Differences between Algorithms.*

Algorithm	Structure	Flexibility and Modification	Known Attacks
DES	Feistel	NO	Brute Force Attack
3DES	Feistel	YES, Extended from 56 to 168 bits	Brute Force Attack, Chosen Plaintext, Known Plaintext
CAST-128	Feistel	YES, 128 and 256 bits	Chosen Plaintext Attack
BLOWFISH	Feistel	YES, 64-448 key length in multiplies of 32	Dictionary Attack
IDEA	Substitution-Permutation	NO	Differential Timing Attack, Key-Schedule Attack
AES	Substitution-Permutation	YES, 256 key length in multiples of 64	Side Channel Attack
RC6	Feistel	YES, 128-2048 key length in multiples of 32	Brute Force Attack, Analytical Attack
RSA	Factorization	YES, Multi Prime RSA, Multi power RSA	Factoring the Public Key

4.4.5. Difference Between Encryption and Decryption

Encryption is the process of converting readable data into unreadable characters to prevent unauthorized access. You treat encrypted data just like any other data. While the Process of converting encoded/encrypted text into a form that is readable and understandable by humans or computers is known as Decryption.

Encryption

Encryption is the process of converting readable data into unreadable characters to prevent unauthorized access. Users treat encrypted data just like any other data. That is, the user

can store it or send it through e-mail message. To read the data, the recipient must decrypt, or decipher it, into a readable form.

In the encryption process, the unencrypted readable data is called plaintext. The encrypted (scrambled) data is called cipher text. An encryption algorithm is a set of steps that can convert readable plaintext into unreadable cipher text. Encryption programs typically use more than one encryption algorithm, along with an encryption key. An encryption key is a programmed formula that the originator of the data used to encrypt the plaintext and the recipient of the data used to decrypt the cipher text.

Some operating systems and e-mail programs allow you to encrypt the contents of files and messages that are stored on the computer. The user can purchase an encryption program, such as Pretty Good Privacy (PGP). A digital signature is an encrypted code that a person, Web site, or organization attaches to an electronic message to verify the identity of the message sender. Digital signatures often are used to ensure that an impostor is not participating in an Internet transaction. That is, digital signatures help to prevent e-mail forgery. A digital signature also can verify that the content of a message has not changed.

Many Web browsers and Web sites use encryption. A Web site that uses encryption techniques to secure its data is known as a secure site. Secure sites often use digital certificates. A digital certificate is a notice that guarantees a user or a Web site is legitimate. A certificate authority (CA) is an authorized person or a company that issues and verifies digital certificates. Users apply for a digital certificate from a CA. The digital certificate typically contains information such as the user's name, the issuing CA's name and signature, and the serial number of the certificate. The information in a digital certificate is encrypted.

Decryption

The process is of converting encoded/encrypted text into a form that is readable and understandable by humans or computers known as Decryption. Decryption refers to the method of un-encrypting the text manually or by using codes and keys. Data is encrypted to secure the information from stealing, and some major companies encrypt data to keep their trade secrets secure. While viewing that data the users need to decrypt the data by using decryption process. If the user wants to view the data, the user can enter passphrase or key and if the user does not have the key, the user may need to decrypt this by using algorithms and cracking it.

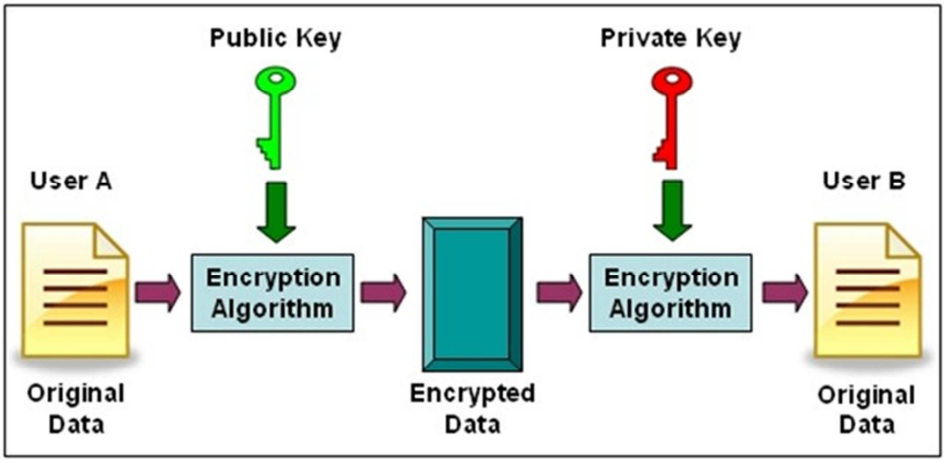


Figure 4. *Key Generation for Encryption in cloud.*

4.5. The Process of Encryption

Encryption is the process of translating plain text data (plaintext) into something that appears to be random and meaningless (cipher text). A symmetric key is used during both the encryption and decryption processes. To decrypt a particular piece of cipher text, the key is used to encrypt the data.

4.5.1. Architecture

(i). This section describes the encryption architecture in MapR.

(ii). MapR uses a mix of approaches to secure the core work of the cluster and the Hadoop components installed on the cluster. For example, nodes in a MapR cluster use different protocols depending on their tasks:

(iii). The FileServer, JobTracker, TaskTracker, NodeManager, and ResourceManager use MapR tickets to secure their remote procedure calls (RPCs) with the native MapR security layer. Clients can use the maprlogin utility to obtain MapR tickets. Web UI elements of these components use password security by default, but can also be configured to use SPNEGO.

(iv). Hive Metastore, Hue, Flume, and Oozie use MapR tickets by default, but can also be configured to use Kerberos.

(v). HBase requires Kerberos for secure communications.

(vi). The MCS Web UI is secured with passwords. The MCS Web UI does not support SPNEGO for users, but supports both password and SPNEGO security for REST calls.

(vii). Servers must use matching security approaches. When an Oozie server, which supports MapR Tickets and Kerberos, connects to HBase, which supports only Kerberos, Oozie must use Kerberos

for outbound security. When servers have both MapR and Kerberos credentials, these credentials must map to the same User ID to prevent ambiguity problems.

4.5.2. Types of Encryption

Two types

(i). Shared secret encryption

(ii). Public key encryption

4.6. Encryption Key Generation

The remainder of the conversation uses a (typically faster) symmetric-key algorithm for encryption. Computer cryptography uses integers for keys. In some cases keys are randomly generated using a random number generator (RNG) or pseudorandom number generator (PRNG).

Pseudorandom encryption key

A pseudorandom number generator (PRNG) is also known as deterministic random bit generator (DRBG). It is an algorithm for generating properties sequence of numbers and the properties sequences of random numbers. The PRNG-generated sequence is not truly random, because it is completely determined by an initial value, called the PRNG's seed (which may include truly random values). Although sequences that are closer to truly random can be generated using hardware random number generators, pseudorandom number generators are important in practice for their speed in number generation and their reproducibility.

A. RNG Method Encrypted Algorithm:

Step-1. Read the plain text as p and key as k, which is the state of the random number generator.

Step-2. Convert each text into its ASCII values.

Step-3. Transform each character of text using the expressions given as: y = p + 2 sin (100) c = y + 10

r k = k + 1. Where p is input text; c is output text; r = random number generated by the state,

"k" of Matlab random number generator;

Step-4. Plot output of the system.

Step-5. Convert integer values into its character values.

Step-6. Read c as output text as cipher text.

B. LCG Method Encryption Algorithm:

Step-1. Read plaintext as p, key as b and length as n.

Step-2. Change the character values of text into its ASCII values.

Step-3. Each ASCII values are transformed into five corresponding values using the following transformations: y = p + sin (b); c = y + r; y is intermediate variable. r is the random numbers generated corresponding to the key; b is state of LCG; c is any variable.

Step-4. Plot „c" obtained from above step.

Step-5. The sequences of numbers in c are then converted into character values.

Step-6. Read the output text (cipher text).

5. Proposed System

Privacy-preserving public auditing for cloud data storage under the mentioned model our protocol design should achieve the security and performance guarantees. For achieving, the objective of the system by using paillier algorithm is used. Symmetrical key generation techniques only concentrated the private authentication. But the asymmetrical key generation techniques is concentrated both private and public. The proposed paillier algorithm is covered the asymmetrical key generation techniques. Therefore, to achieve the objective of the system, paillier algorithm has been used to generate the efficient result in cloud. The purpose of the algorithm is used for encryption the fact that RSA is a (conjunction) trapdoor function.

Symmetric algorithms: (also called "secret key") use the same key for both encryption and decryption; asymmetric algorithms: (also called "public key") use different keys for encryption and decryption. The distinguishing technique used in public key cryptography is the use of asymmetric key algorithms, where a key used by one party to perform encryption is not the same as the key used by another in decryption. Each user has a pair of cryptographic keys – a public encryption key and a private decryption key.

The proposed system is implemented on an Intel core i5 processor system running at 2.20 GHz, 3GB RAM using Java and Ulteo OVD virtual desktop for building cloud environment. The implemented system consists of 5 modules: User registration, encrypt and upload, file sharing, decrypt and download and verification auditing.

User Registration

The registration function allows users to create secure account. Here the user enters his/her information necessary for signing up like user's name, password, mobile no and email-address. The validations and required fields are effectively handled. Each user will be provided his/her own space on cloud.

Encrypt and Upload

After registering, the user may login into the system. Every user is provided space on cloud where they may upload their files. The encrypt function mentioned below

```
#include <stdint. h>
void encrypt (uint32_t* v, uint32_t* k)
{
   uint32_t v0=v [0], v1=v [1], sum=0, i=0; /* set up */
   uint32_t delta=0x9e3779b9; /* a key schedule constant
*/
   uint32_t k0=k [0], k1=k [1], k2=k [2], k3=k [3]; /*
cache key */
   while (i<=32)
   { /* basic cycle start */
       sum += delta;
       v0 += ((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) +
     k1);
       v1 += ((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) +
     k3); i++
      } /* end cycle */
v [0]=v0; v [1]=v1;
}
```

Public-key encryption:

This idea omits the need for a “courier" to deliver keys to the recipients over another secure channel before transmitting the originally-intended message. In RSA, encryption keys are public, while the decryption keys are not, so only the person with the correct decryption key can decipher an encrypted message. Everyone has their own encryption and decryption keys. The keys must be made in such a way that the decryption key may not be easily deduced from the public encryption key.

Digital signatures:

The receiver may need to verify that a transmitted message actually originated from the sender (signature), and it didn't just come from there (authentication). This is done by the sender's decryption key, and the signature can later be verified by anyone, using the corresponding public encryption key. Signatures therefore cannot be forged. Also, no signer can later deny the message.

This is not only useful for electronic mail, but also for other electronic transactions and transmissions, such as fund transfers. The security of the RSA algorithm has so far been validated, since no known attempts to break it and yet it has been successful, mostly due to the difficulty of factoring large numbers

n = pq, where p and q are large prime numbers.

Public-key cryptosystems:

Each user has their own encryption and decryption procedures, E and D, with the former in the public and the latter kept secret. These procedures are related to the keys, which, in RSA specifically, are sets of two special numbers. Of course start out with the message itself, symbolized by M, which is to be “encrypted". There are four procedures that are specific and essential to a public-key cryptosystem.

a) Deciphering an enciphered message gives you the original message, specifically

$$D(E(M))=M \tag{1}$$

b) Reversing the procedures still returns M:

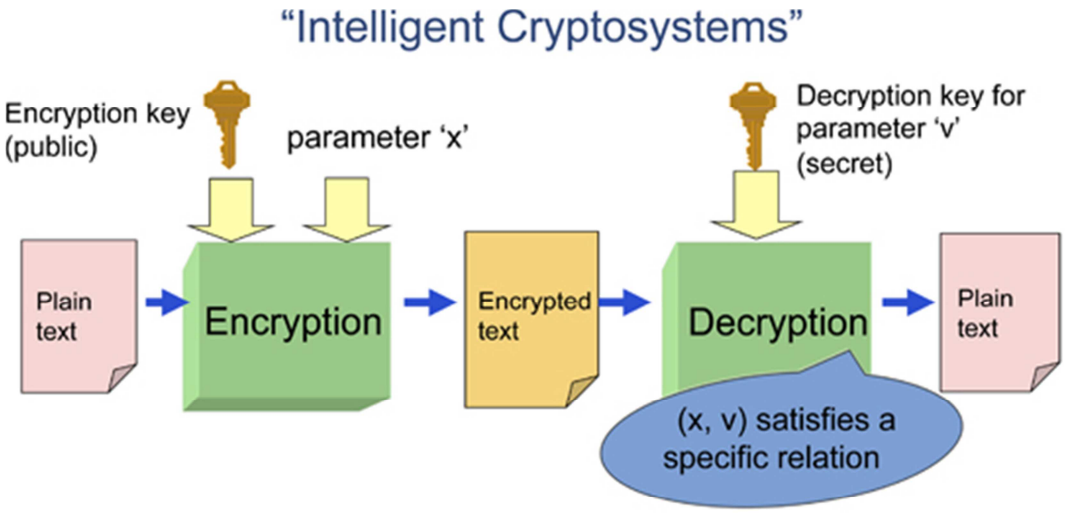
$$E(D(M))=M \tag{2}$$

c) E and D are easy to compute.

d) The publicity of E does not compromise the secrecy of D, meaning you cannot easily figure out D from E.

With a given E, we are still not given an efficient way of computing D. If C = E (M) is the cipher text, then trying to figure out D by trying to satisfy an M in E (M) = C is unreasonably difficult: the number of messages to test would be impractically large.

An E that satisfies (a), (c), and (d) is called a “trap-door one-way function" and is also a “trap-door one-way permutation". It is a trap door because since it's inverse D is easy to compute if certain “trap-door" information is available, but otherwise hard. It is one-way because it is easy to compute in one direction, but hard in the other. It is a permutation because it satisfies (b), meaning every cipher text is a potential message, and every message is a cipher text of some other message. Statement (b) is in fact just needed to provide “signatures". Now turn to specific keys, and imagine users A and B on a two-user public-key cryptosystem, with their keys: E_A, E_B, D_A, D_B.



	Encryption parameter ‘x’	Decryption parameter ‘v’	Representative application
Use case 1	Attribute information	Conditional expression Can be decrypted if attribute ‘x’ satisfies conditional expression ‘v’	Contents delivery, etc.
Use case 2	Conditional expression	Attribute information Can be decrypted if attribute ‘v’ satisfies conditional expression ‘x’	Browsing of important in-house documents, etc.

Figure 5. *Intelligent Cryptosystems.*

Privacy:

Encryption, which is now a ubiquitous way of assuring a message is delivered privately, so no intruder can bypass the cipher text, which is essentially white noise. Without property (d), however, an encryption process is still not public-key, such as the NBS standard. It requires keys to be delivered privately through another secure “courier", which is an extra process that would deem NBS, for example, as slow, inefficient, and possibly expensive. Thus, RSA is a great answer to this problem. The NBS standard could provide useful only if it was a faster algorithm than RSA, where RSA would only be used to securely transmit the keys only. Thus, an efficient computing method of D must be found, so as to make RSA completely stand-alone and reliable. For it to be reliable, it would have to use simple arithmetic, which is easier to compute (a requirement of property (c)) on a general-purpose computer than are bit manipulations, where better hardware is used, where perhaps NBS would be better.

Signatures:

For complete assurance that the message originatede from a sender, and was not just sent through him by a third party who may have used the same encryption key (that of the receiver), need a digital signature to come with the message. This has obvious implications of importance in real-life applications.

$$DB\ (M)=S \tag{3}$$

Then encrypt S with encryption key

$$Ea\ (S) = Ea\ (Db\ (M)) \tag{4}$$

This way can assure only decrypt the document and gets the signature by

$$D_A\ (E_A(D_B(M)) = S.$$

Since only decryption key could compute the signature. The messages need not be sent separately, since deduce it from the signature itself by using publicly available encryption key,

Formally

$$E_B(S) = E_B\ (D_B(M)) = M.$$

Since S depends on M, and the encrypted transmission sent depends on S, the transmission that depends on both the message and the signature, so both can be deduced from the transmitted document. This brilliantly assures the message could not be modified (if needed to be presented to, say, a judge"), since a modified M in the form of M^0 would have to generate a signature

$$S^0 = D_B\ (M^0)$$

Which is impossible, since does not know D_B by property (d). So not only possess proof that signed the message and indeed sent it, but she also cannot modify M nor forge a signature for any other message.

A signature just need to assure it came from the public File (PF) itself. Every time a user joins a network, everybody gets a securely sent copy of the most recently updated PF, which is stored on their system, and they never have to look it up. Anyone trying to send a message pretending to be in the public file would not have the appropriate signature, and would be singled out as an intruder.

Applications, predictions, hardware implementation:

This has applications to electronic fund transmissions as well. Financial information needs to be secure, and checks can be electronically signed with RSA. Further measures have to be taken, such as implementing unique check numbers that allow a check with this certain number transmittable cashable only once.

In fact, such a system can be applied to any electronic system that needs to have a cryptosystem implemented. In 1978 RSA paper, the authors of RSA predicted a secure email world to evolve and for RSA to be used to encrypt a live telephone conversation. Now, these things are indeed a part of more than just daily life because of RSA.

The encryption device must not be the direct buffer between a terminal and the communications channel. Instead, it should be a hardware subroutine that can be executed as needed, since it may need to be encrypted/decrypted with several sequences of keys, so as to assure more privacy and/or more signatures.

File sharing:

The data owner may share the outsourced files with other users in cloud using the share module. The secret key generated during encryption is also mailed to the shared user in order to grant them access to the shared file.

Decrypt and download:

The data owner may share the outsourced files with other users in cloud using the share module. The secret key generated during encryption is also mailed to the shared user in order to grant them access to the shared file. The shared user may download the file, make changes and again upload the file. In such a case, TPA informs the original data owner of that file about the latest modifications done by a shared user.

Decryption Algorithm:

```
void decrypt (uint32_t* v, uint32_t* k)
{
   uint32_t v0=v [0], v1=v [1], sum=0xC6EF3720, i=0; /*
set up */
   uint32_t delta=0x9e3779b9; /* a key schedule constant
*/
   uint32_t k0=k [0], k1=k [1], k2=k [2], k3=k [3]; /*
cache key */
   while (i<32)
   { /* basic cycle start */
       v1 -= ((v0<<4) + k2) ^ (v0 + sum) ^ ((v0>>5) + k3);
       v0 -= ((v1<<4) + k0) ^ (v1 + sum) ^ ((v1>>5) + k1);
       sum -= delta; i++;
    } /* end cycle */
    v [0]=v0; v [1]=v1;
    }
```

The math of the method:

So far, expect to make E and D easy to compute through simple arithmetic. Now represent the message numerically, so that can perform these arithmetic algorithms on it. Now lets represent M by an integer between 0 and n 1. If the message is too long, sparse it up and encrypt separately. Let e; d; n be positive integers, with (e; n) as the encryption key, (d; n) the decryption key, n = pq.

Now encrypt the message by raising it to the e^{th} power modulo n to obtain C, the cipher text. We then decrypt C by raising it to the d^{th} power modulo n to obtain M again.

Formally obtain these encryption and decryption algorithms for E and D:

$$CE (M) Me (mod n) \quad (5)$$

$$MD (C) Cd (mod n) \quad (6)$$

Note that preserving the same information size, since M and C are integers between 0 and n 1, and because of the modular congruence. Also note the simplicity of the fact that the encryption/decryption keys are both just pairs of integers, (e; n) and (d; n). These are different for every user, and should generally be subscripted, but we'll consider just the general case here.

Now comes the question of creating the encryption key itself. First, choosing two “random" large primes p and q, multiply and produce n = pq. Although n is public, it will not reveal p and q since it is essentially impossible to factor them form n, and therefore will assure that d is practically impossible to derive from e.

Now we want to obtain the appropriate e and d. We pick d to be a random large integer, which must be co-prime to (p 1) (q 1), meaning the following equation has to be satisfied:”gcd” means greatest common divisor.

$$gcd (d; (p 1) (q 1)) = 1 \quad (7)$$

The reason want d to be co-prime to (p 1) (q 1) is peculiar. It will not show the “direct motivation" behind it; rather, it will become clear why that statement is important when show towards the end of this section that it guarantees (1) and (2). It will want to compute e from d, p, and q, where e is the multiplicative inverse of d. That means we need to satisfy

$$e.d = 1 (mod \varphi (n)) \quad (8)$$

Here introduce the Euler totient function (n), whose output is the number of positive integers less than n which are co-prime to n. For primes p, this clearly becomes (p) = p 1. For n, obtain by elementary properties of the totient function, that

$$\Phi (n) = \Phi (p) \Phi (q)$$

$$= (p -1) (q -1)$$

$$= n \Phi (p + q) + 1$$

From this equation, can substitute (n) into equation (7) and obtain which is equivalent to for some integer k.

$$e. d1 (mod \Phi (n))$$

$$e d = k. \Phi (n) + 1 \quad (9)$$

By the laws of Φ (n) modular arithmetic, the multiplicative inverse of a modulo m exists if and only if a and m are co-prime. Indeed, since d and (n) are co-prime, d has a multiplicative inverse e in the ring of integers modulo Φ (n). So far, can safely assured the following:

$$D (E (M)) (E (M))^d (M^e)^d (mod n) = M^{e d} (mod n)$$

$$E (D (M)) (D (M))^e (M^d)^e (mod n) = M^{e d} (mod n)$$

$$M \Phi^{(n)} = 1 (mod n) \quad (10)$$

Also, since e.d = k. Φ (n) + 1, we can substitute into the above equations and obtain

$$_M e. d = _M k- \Phi (n)+1_{(mod\cdot n)}$$

Clearly represent that to equal M. To prove this, will need an important identity due to Euler and Fermat: for any integer M co-prime to n

Since previously specified that 0 M < n and know that M would not be co-prime to n if and only if M was either p or q, of the integers in that interval. Therefore, the chances of M happening to be or q are on the same order of magnitude as 2=n. This means that M is almost definitely relatively prime to n, therefore equation (9) holds and, using it, evaluate:

$$Me. d; Mk- \Phi (n)+1; (M \Phi (n)) kM; 1kM; (mod n) = M$$

It turns out this works for all M, and in fact see that (1) and (2) hold for all M; 0 M < n. Therefore E and D are inverse permutations.

Algorithms:

(i). Efficient encryption and decryption operations.

(ii). The authors of RSA claim that “computing Me (mod n) requires at most 2 log2 (e) multiplications and 2 log2 (e) divisions" if use their procedure below. It is important for us to know the amount of steps it would take a computer to encrypt the message so can see if a method is fast and efficient, or not. Now exponentiation by repeated squaring and multiplication".

Step 1. Let ekek 1:::e1e0 be the binary representation of e.

Step 2. Set the variable C to 1.

Step 3. Repeat steps 3a and 3b for i = k; k1;:::; 0:

Step 3a. Set C to the remainder of C2 when divided by n.

Step 3b. If ei = 1 then set C to the remainder of C M when divided by n.

Step 4. Halt. Now C is the encrypted form of M.

(i). There are more efficient procedures out there, but this one is good too. Also, since decryption follows the same identical procedure as encryption and can implement the whole operation on a few integrated chips.

(ii). According to the authors of RSA, the encryption time per block increases no faster than the cube of the number of digits in n.

Verification auditing:

In order to authenticate the integrity of the user's uploaded data, the TPA is granted access to the system. The TPA validates the integrity of the cloud data files on remote server on behalf of cloud user itself. TPA verifies the legitimacy of data using secret hash key sent by the cloud user. If the secret hash key matches with hash key in the cloud server, the verification proves to be successful, thus implying that the data files has not been modified. However, if the verification is unsuccessful, an email is dispatched to the data owner of the file informing about the last modifications done to his file.

Paillier Algorithm:

The Paillier cryptosystem is invented and named after Pascal Paillier in 1999, is a probabilistic asymmetric algorithm for public key cryptography. The problem of computing n-th residue classes is believed to be computationally difficult. The decisional composite residuosity assumption is the intractability hypothesis upon which this cryptosystem is based. The scheme is an additive homomorphic cryptosystem; this means that, given only the public-key and the encryption of m1 and m2, one can compute the encryption of m1+m2.

(i). Select two large primes, p and q.
(ii). Calculate the product n=p x q, such that gcd (n, Φ (n)) = 1, where Φ (n) is Euler Function.
(iii). Choose a random number g, where g has order multiple of n or gcd (L (gλ mod n2), n) = 1, where L (t)= (-1) / n and λ (n)=lcm (p-1, q-1).
(iv). The public key is composed of (g, n), while the private key is composed of (p, q, λ).
(v). The Encryption of a message m < n is given by:
(vi). c=gmrn mod n2
(vii). The Decryption of ciphertext c is given by:
(viii). m=(L (gλ mod n2)/L (gλ mod n2)) mod n.

Finding large prime numbers.

Finding n is the first step to the entire process. The number n will be revealed in the encryption and decryption keys, but the numbers p and q, whose product make up n, will not be explicitly shown. They are essentially impossible to derive from n, in fact, especially if we pick, say, 100-digit primes p and q, which would make a 200-digit n. Each user needs to privately choose his own two large prime numbers p and q. To do this, need to generate, say, random odd 100-digit numbers until a prime is found an will have to test each number, and according to the prime number theorem, there will be about (ln 10^{100})/2 = 115 number to test.

To test a large b for primality, can use an algorithm due to Solovay and Strassen. First, pick a random number from a uniform distribution on 1; b 1 and test whether

$$\gcd(a; b) = 1 \text{ and } J(a; b) = a^{(b\ 1)\ 2} \pmod{b}; \qquad (11)$$

The Jacobi symbol is only defined when a is an integer and b is a positive odd integer. Also, J (a; b) is 0 if gcd (a; b) 6= 1 and 1 if gcd (a; b) = 1. Equation (10) is always true if b is prime, otherwise (if b is composite), (10) will have a chance of being false of over 50%. If (10) is true 100 times for randomly chosen a's, then b is almost certainly prime, with a chance of being composite of 1 in 2^{100}. If accidentally a composite were used for p or q in the process, the recipient would see "junk" and realize the decryption wasn't done correctly. Now present an efficient program for computing J (a; b)

J (a;b) = if a=1 then 1 else
if is even then j (a12;b). (1) (b2 l) / 8
else j (b (mod a);a). (1)^(a l) (b 1) /4

To protect against sophisticated factoring algorithms, p and q should differ in length by a few digits, gcd (p 1; q 1) should be small, and both (p 1) and (q 1) should contain large prime factors. To assure the latter, we generate a large random prime number u and take the first prime in the sequence i u + 1; i = 2; 4; 6;:::. This process would be very fast on a computer. When the authors of RSA published their article, on a high-speed computer, testing a 100-digit number for primality would take several seconds, while finding the next prime would take around a minute and a half. We could also find large primes by taking a number whose factorization we know, add 1 to it, and test for primality. If we get a number we think is prime, we could potentially prove that it is prime by using the factorization of (p 1)

How secure is RSA?

The RSA algorithm is indeed among the strongest, but can it withstand anything? Certainly nothing can withstand the test of time. In fact, no encryption technique is even perfectly secure from an attack by a realistic cryptanalyst. Methods such as brute-force are simple but lengthy and may crack a message, but not likely an entire encryption scheme. We must also consider a probabilistic approach, meaning there's always a chance someone may get the \one key out of a million". So far, we don't know how to prove whether an encryption scheme is unbreakable. If we cannot prove it, we will at least see if someone can break the code. This is how the NBS standard and RSA were essentially certified. Despite years of attempts, no one has been known to crack either algorithm. Such a resistance to attack makes RSA secure in practice.

Factoring large numbers is not provably hard, but no algorithms exist today to factor a 200-digit number in a reasonable amount of time. Fermat and Legendre have both contributed to this field by developing factoring algorithms, though factoring is still an age-old math problem. This is precisely what has partially "certified" RSA as secure.

To show that RSA is secure, will consider how a cryptanalyst may try to obtain the decryption key from the public encryption key, and not how an intruder may attempt to "steal" the decryption key. This should be taken care of as one would protect their money, through physical security methods. The authors of RSA provide an example: the encryption device (which could be, say, a set of integrated chips within a computer), would be separate from the rest of the system. It would generate encryption and decryption keys, but would not print out the decryption key, even for its owner. It would, in fact, erase the decryption key if it sensed an attempted intrusion.

Factoring:

Since knowing the factors of n would give away (n) and therefore d, a cryptanalyst would break the code if factored n. However, factoring numbers has practically proven to be far harder than determining primality or compositeness. Nonetheless, many factoring algorithms are around. The following table is the one the authors of RSA presented in 1978. They assume an operation in the Schroeppel factoring algorithm takes one microsecond to compute, and present the following data for various lengths of n:

Table 2. *RSA factoring algorithm time duration for digits*

Digits	Number of operations	Time
50	1.4×10^{10}	3.9 hours
75	9.0×10^{12}	104 days
100	2.3×10^{15}	74 years
200	1.2×10^{23}	3.8×10^{9} years
300	1.5×10^{29}	4.9×10^{15} years
500	1.3×10^{39}	4.2×10^{25} years

6. Conclusion

The main objective and aim of this paper is that to study about the integrity verification strategy for outsourced data. For that the proposed scheme developed and which combines the encrypting, decryption mechanism and its algorithms has been discussed and it's implemented to get efficient result. To achieve the efficient result RSA and paillier algorithms have been used. For key generation asymmetric method has been implemented.

References

[1] Sarah sheikh and Deepali vora, "Secure cloud auditing over encrypted data", IEEE transactions on information forensics and security, Vol. 11, No. 6, Mar 2017, PP: 1-5.

[2] Kalyani sonaware and Rahul dagade, "A survey on multi-key word ranked search over encrypted cloud data with multiple data owners", International journal of computer applications, Vol. 162, No. 11, Mar 2017, PP:9-12.

[3] Swapnali and Snagita chaudhari, "Third party public auditing scheme for cloud storage", 7th International conference on communication, computing and virtualized 2016, PP: 69-76.

[4] Nitin nagar and Ugrasen suman, "Reliable & enhanced third party auditing in cloud server data storage", International journal of security and its applications, Vol. 11, No. 7, 2017, PP: 59-71.

[5] Mahesh kumar, "Encrypted bigdata using AES deduplication in cloud storage", International journal of engineering and computer science, Vol. 6, No. 7, July2017, PP: 21889-21894.

[6] Sweta k. parmer and K. C. Dave "Implementation of data encryption & decryption algorithm for information security", International journal of advanced in science engineering and technology, Vol. 1, No. 2, Oct2013, PP: 7-10.

[7] Pitchaiah, Philemon Daniel & Praveen, "Implementation of advanced encryption standard algorithm", International journal of engineering research, Vol. 3, No. 3, 2012, PP: 1-6.

[8] Vaishali Patil & Archana Lomte, "Implementation of privacy –preserving public auditing and secure searchable data cloud storage", Vol. 3, No. 7, 2017, PP: 65-72.

[9] Karthika priya dharshini, Viji & Saravanan, "Seclusion search over encrypted data in cloud storage services", Vol. 4, No. 3, Mar 2015, PP: 27-34.

[10] Evgeny Milanov, "The RSA Algorithm", 3, June 2007, PP:1-11.

Biography

Yesu Ragavi Kannadasan is completed her under graduation course Bachelor of Computer Applications in Vivekanandha College of Arts and Sciences for women in the year 2016. She has attended 15 National and International level workshops and attended more than 13 National and International seminars. She has presented a paper in International conference and published a paper in International Journal. Her research area is Cloud Computing and Computer Networks. Currently she is pursuing her Master of Computer Applications degree in Vivekanandha College of Arts and Sciences for women and it is affiliated to Periyar University.

Rajagopal Devarajan, he has completed his Bachelor of Computer Science degree and completed his Master of Computer Applications degree in Periyar University in the year 2003 and 2006 respectively. He has completed his Master of Philosophy in PRIST University in the year of 2012. He has 3 Years and 10 Months Experience in the field of Software Developing and 7 Years 6 months Experience in Teaching. He has published 14 International Journal papers. Currently he is working as an assistant professor in the department of MCA, Vivekanandha College of Arts and Sciences for women, Tiruchengode.