International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

# Provide Security to User Data on Free Cloud

**Namdev S.Choure[1], Mahesh A. Kale[2], Omkar B.Pandit[3], Pranav Alhat[4], Prof.Shrikant Dhamdhere[5]**

*[1,2,3,4] Computer Engineering, PGMCOE (INDIA)*

*[5]HOD of Computer Engineering at PGMCOE(INDIA)*

## ABSTRACT

*Cloud computing is a new computing model which is widely emerging technology in the recent years. It is used by many organization as well as Single Users. However, a security issue has been always obstacle to the use of computing outsourcing. The cloud environment is considered untrusted as*

*it is accessed through Internet. Therefore people have security concerns on data stored in cloud environment like data confidentiality, data integrity and user authentication. We proposed a new approach for securely storing our data in cloud with the help of cryptographic algorithm for data confidentiality, for integrity checking mechanism by which we can check whether data integrity is preserved or not at the time of data retrieval we used hash function and user authentication using unique password (OTP) or number is obtained through the users personal email via text*

***Keywords* : *Cryptography, Encryption, Hashing, AES, SHA-1*.**

## INTRODUCTION

keeping our personal information private is becoming more difficult. The truth is, highly classified details are becoming more available to public databases, because we are more interconnected than ever. Our data is available for almost anyone to sift through due to this interconnectivity. This creates a negative stigma that the use of technology is dangerous because practically anyone can access one's private information for a price. Common internet users may be unaware of cybercrimes, let alone what to do if they fall victim of cyber attacks. While online data storage services claim your data is encrypted, there are no guarantees. We can not trust third party free cloud because our confidential data is misused by them and normally we can not recognize that our downloaded data from free cloud is modified or not. People are usually more concerned about there data so that why we proposed a system where we provide confidentiality and integrity to users data at the time of storing their data on free cloud. In Confidentiality we uses encryption technique and integrity we used hash function. For encryption we used AES10 and for the hashing we used SHA-1 algorithm.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

## Proposed System

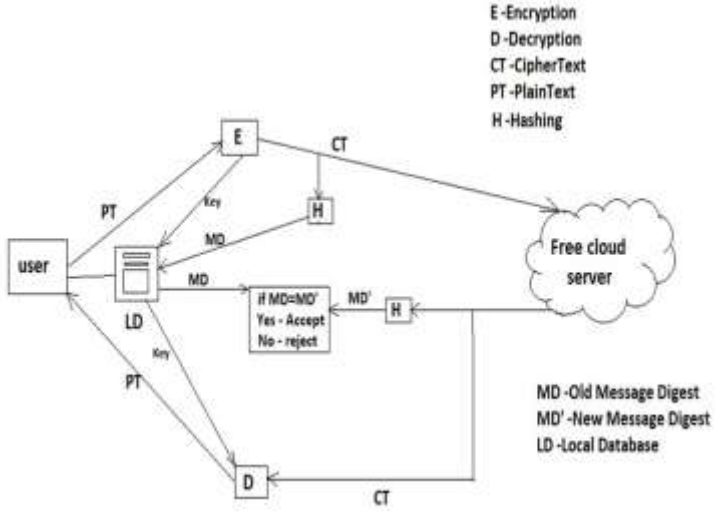



Fig. Architecture Diagram

Above figure shows architecture representation of our system, whenever users want store data on free cloud at that time our system first encrypt that data, for encryption we used a AES-10 algorithm and then key store into our Local Database that same is use for decryption when encrypted file download in future. After performing encryption we get cipher text or encrypted data of user. After this we will calculate message digest of that cipher text by using SHA-1 algorithm and store this message digest to our Local Database. After that store the cipher text into free cloud.

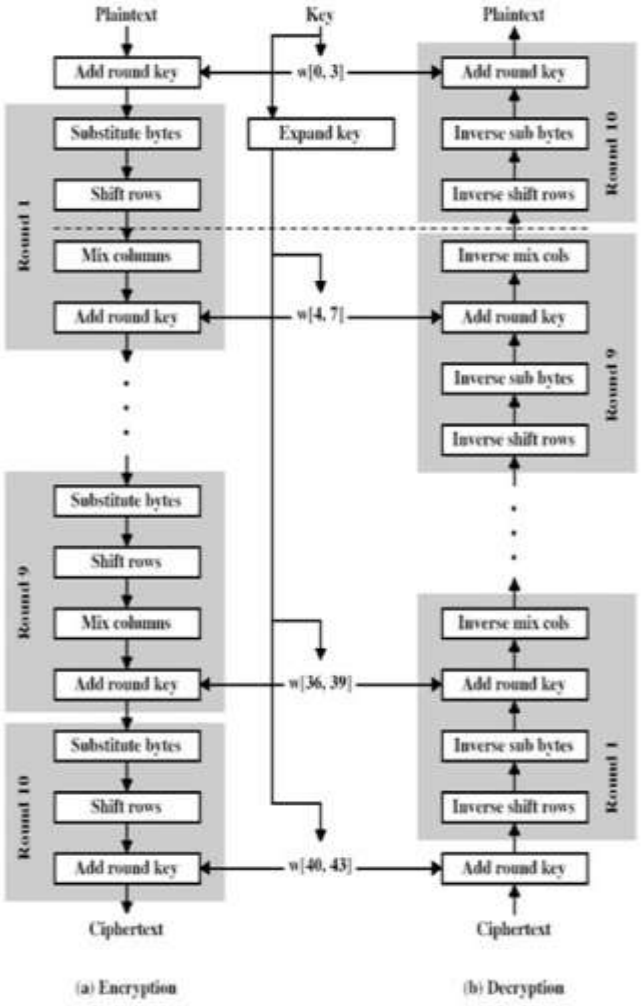
So after storing the data on cloud user want to download that data from cloud. Then it will download that data, then it will again calculate the new message digest of user downloaded data and then compare this message digest with old message digest which was store in Local Database. If this both message digest is equal then it will sends acknowledgement to user that their data is not get modified else it will sends acknowledgement that their data get modified and if data get modified it will reject that data, if data get not modified it will accept this data and decrypt that data by using the key that we will store in Local Database. After decryption process done user get their original data securely.

## ALGORITHMS

### 1) AES-10

AES stands for Advanced Encryption Standard. AES was created by Vincent Rijmen and Joan Daeman. Since of the little key length the DES is never again considered as safe for todays applications. AES concoct key length 128bit utilizing the symmetric square figure. AES calculation isn't just for security yet in addition for incredible speed.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354




The algorithm begins with an Add round key stage followed by 9 rounds of four stages and a tenth round of three stages. This applies for both encryption and decryption with the exception that each stage of a round the decryption algorithm is the inverse of its counterpart in the encryption algorithm.

The four stages are as follows:

1. Substitute bytes
2. Shift rows
3. Mix Columns
4. Add Round Key

The tenth round simply leaves out the Mix Columns stage. The first nine rounds of the decryption algorithm consist of the

following:

1. Inverse Shift rows
2. Inverse Substitute bytes

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

3. Inverse Add Round Key

4. Inverse Mix Columns

Again, the tenth round simply leaves out the Inverse Mix Columns stage. Each of these stages will now be considered in more detail.

Fig 1: AES-10 Algorithm Flowchart

## 1.1) Substitute Bytes

This stage (known as SubBytes) is simply a table lookup using a 16*16 matrix of byte values called an s-box. This matrix consists of all the possible combinations of an 8 bit sequence (28 = 16 16 = 256). How ever, the s-box is not just a random permutation of these values and there is a well desingned method for creating the s-box tables. The designers of Rijndael showed how this was done unlike the s-boxes in DES for which no rationale was given. We will not be too concerned here how

the s-boxes are made up and can simply take them as table lookups. Again the matrix that gets operated upon throughout the encryption is known as state. We will be concerned with how this matrix is selected in each round.

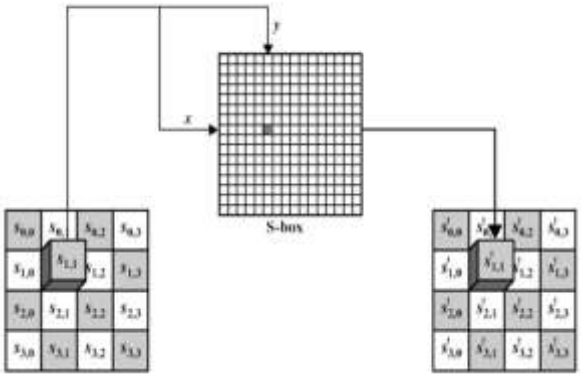



Fig 1.1 Substitute Bytes

## 1.2) Shift Row Transformation

This stage (known as Shift Rows) is shown in fig 1.2. This is a simple permutation an nothing more. It works as follow:

A. The first row of state is not altered.

B. The second row is shifted 1 bytes to the left in a circular manner.

C. The third row is shifted 2 bytes to the left in a circular manner.

D. The fourth row is shifted 3 bytes to the left in a circular manner.

The Inverse Shift Rows transformation (known as InvShiftRows) performs these circular shifts in the opposite direction for each of the last three rows (the first row was unaltered to begin with).

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

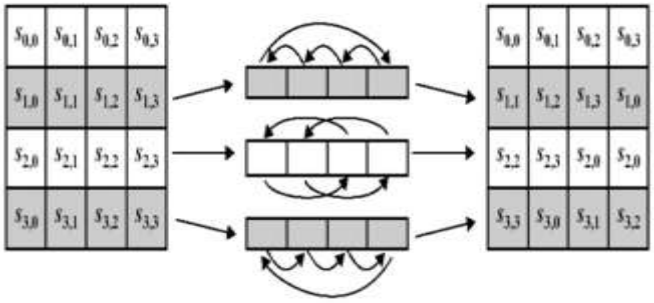


Fig 1.2 Shift Row Transformation

## 1.3) Mix Column Transformation

This stage (known as Mix Column) is basically a substitution but it makes use of arithmetic of GF(28). Each column is operated on individually. Each byte of a column is mapped into a new value that is a function of all four bytes in the column. The transformation can be determined by the following matrix multiplication on state.

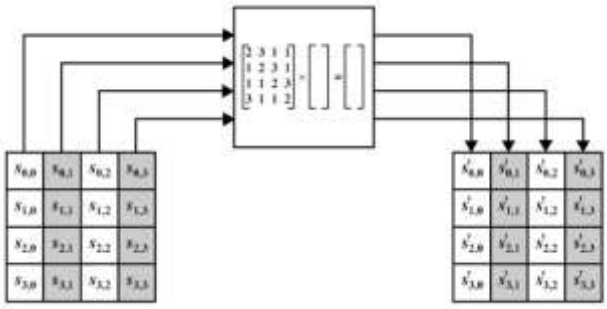


Fig 1.3 Mix Column Transformation

## 1.4) Add Round Key Transformation

In this stage (known as AddRoundKey) the 128 bits of state are bitwise XORed with the 128 bits of the round key. The operation is viewed as a columnwise operation between the 4 bytes of a state column and one word of the round key. This transformation is as simple as possible which helps in efficiency but it also effects every bit of state.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

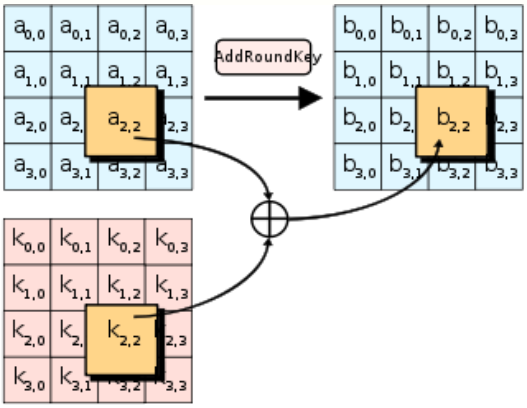



Fig 1.4 Add Round Key Transformation

## 1.5) AES Key Expansion

The AES key expansion algorithm takes as input a 4-word key and produces a linear array of 44 words. Each round uses 4 of these words as shown in Each word contains 32 bytes which means each sub key is 128 bits long. Figure show pseudo code for generating the expanded key from the actual key.

The key is copied into the first four words of the expanded key. The remainder of the expanded key is filled in four words at a time. Each added word w[i] depends on the immediately preceding word, w[i 1], and the word four positions back w[i 4]. In three out of four cases, a simple XOR is used. For a word whose position in the w array is a multiple of 4, a more complex function is used. Figure illustrates the generation of the first eight words of the expanded key using the symbol g to represent that complex function. The function g consists of the following sub functions

1. Rot Word performs a one-byte circular left shift on a word. This means that an input word [b0, b1, b2, b3] is transformed into [b1, b2, b3, b0].

2. SubWord performs a byte substitution on each byte of its input word, using the s-box described earlier.

3. The result of steps 1 and 2 is XORed with round constant, Rcon[j].The round constant is a word in which the three rightmost bytes are always

4. Thus the effect of an XOR of a word with Rcon is to only perform an XOR on the leftmost byte of the word. The round constant is different for each round and is defined as Rcon[j] =(RC[J], 0,0,0), with RC[1]= 1, RC[j]=2 RC[j 1] and with multiplication defined over the field GF(28).The key expansion was designed to be resistant to known cryptanalytic attacks. The inclusion of a round-dependent round constant eliminates the symmetry, or similarity, between the way in which round keys are generated in different rounds.

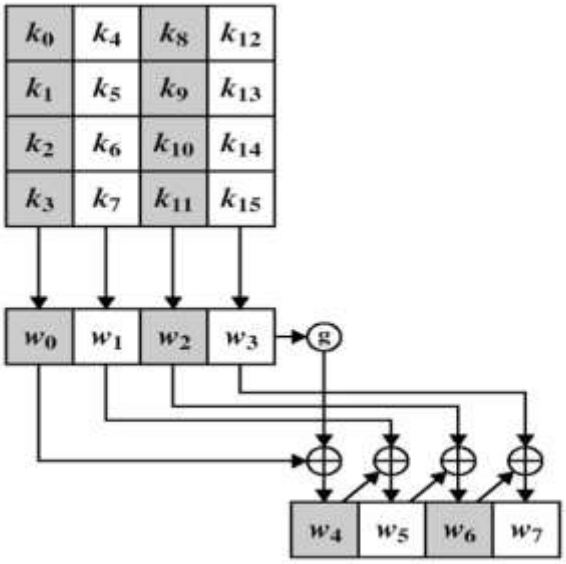



Fig 1.5 AES Key Expansion

## 1.6) Decryption process

Decryption is a Inverse process of Encryption

In Decryption process also perform four operation that is

1. Inverse shift rows
2. Inverse sub bytes
3. Inverse mix columns
4. Add Round Key

## 2) SHA-1

SHA-1 (Secure Hash Algorithm 1) is a cryptographic hash function which takes an input and produces a 160-bit(20-byte) hash value known as a message digest typically rendered as a hexadecimal number, 40 digits long. It was designed by the United States National Security Agency, and is a U.S.Federal Information Processing Standard. One iteration within the SHA-1 compression function: A, B, C, D and E are 32-bit words of the state; F is a nonlinear function that varies; left shift n denotes a left bit rotation by n places;n varies for each operation; Wt is the expanded message word of round t; Kt is the round constant of round t; Addition denotes addition modulo 2^32.

International Journal of Advance Research in Science and Engineering
Volume No.07, Special Issue No.04, April 2018
www.ijarse.com
IJARSE
ISSN: 2319-8354

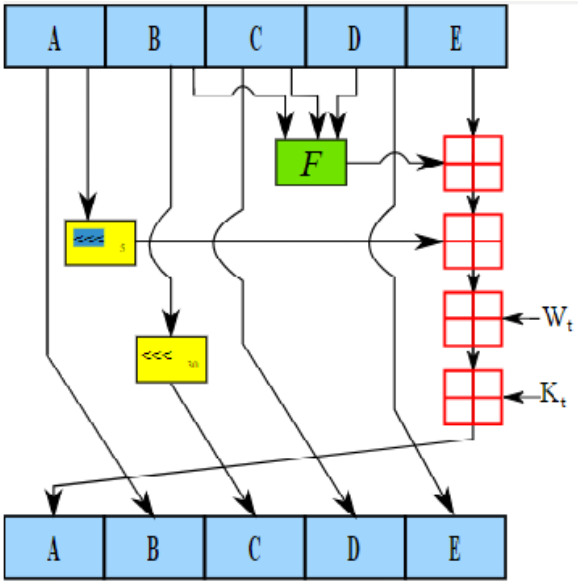



Fig 2: SHA-1 Algorithm

## SHA-1 pseudo code

***Initialize variables:***

h0 = 0x67452301
h1 = 0xEFCDAB89
h2 = 0x98BADCFE
h3 = 0x10325476
h4 = 0xC3D2E1F0

ml = message length in bits (always a multiple of the number of bits in a character).

***Pre-processing****:*
append the bit '1' to the message e.g. by adding 0x80 if message length is a multiple of 8 bits.append $0 \leq k < 512$ bits '0', such that the resulting message length in *bits* is congruent to $-64 \equiv 448 \pmod{512}$ append ml, the original message length, as a 64-bit big-endian integer. Thus, the total length is a multiple of 512 bits.
***Process the message in successive 512-bit chunks****:*
break message into 512-bit chunks for each chunk break chunk into sixteen 32-bit big-endian words w[i], $0 \leq i \leq 15$

***Extend the sixteen 32-bit words into eighty 32-bit words:***

**for** i **from** 16 to 79

w[i] = (w[i-3] **xor** w[i-8] **xor** w[i-14] **xor** w[i-16]) **leftrotate** 1

***Initialize hash value for this chunk:***

a = h0

b = h1

c = h2

d = h3

e = h4

***Main loop:***

**for** i **from** 0 **to** 79

**if** 0 ≤ i ≤ 19 **then**

f = (b **and** c) **or** ((**not** b) **and** d)

k = 0x5A827999

**else if** 20 ≤ i ≤ 39

f = b **xor** c **xor** d

k = 0x6ED9EBA1

**else if** 40 ≤ i ≤ 59

f = (b **and** c) **or** (b **and** d) **or** (c **and** d)

k = 0x8F1BBCDC

**else if** 60 ≤ i ≤ 79

f = b **xor** c **xor** d

k = 0xCA62C1D6

temp = (a **leftrotate** 5) + f + e + k + w[i]

e = d

d = c

c = b **leftrotate** 30

b = a

a = temp

***Add this chunk's hash to result so far:***

h0 = h0 + a

h1 = h1 + b

h2 = h2 + c

h3 = h3 + d

h4 = h4 + e

***Produce the final hash value (big-endian) as a 160-bit number:***

hh = (h0 **leftshift** 128) **or** (h1 **leftshift** 96) **or** (h2 **leftshift** 64) **or** (h3 **leftshift** 32) **or** h4

## Result Analysis

Cyber crimes cases registered in the country have grown in the last seven years, from 1791, 2876, 4356, 7201 to 8045 during 2011, 2012, 2013, 2014 respectively and the number rising from8045 to 12,317 during 2015 and 2016 respectively.


## Conclusion

The system has been developed for enhance security to users confidential data storing on free cloud.So here we used cryptographic algorithm and hash function for giving security to user data. Here we used AES10 algorithm for encryption and SHA1 algorithm for hashing. It provide users confidential data ,data integrity, and user authentication using unique password OTP.So that user confidential data will not be misused.

## Acknowledgments

It gives us great pleasure in writing the paper on ”Provide Security To Data on Free Cloud” . We would like to take this opportunity to thank my internal guide Prof. Shrikant Dhamdhere for giving us all the help and guidance we needed. We are really grateful to them for their kind support. Their valuable suggestions were very helpful. Finally we express our sincere thanks to Prof. Shrikant Dhamdhere for giving us all the help and guidance that We needed, and all those who helped us directly or indirectly in many ways in completion of this project work.

## REFERENCE

[1]. Neha, Mandeep Kaur,M.Tech. Student, Department of Computer Engineering, RBIEBT, Mohali, India "Enhanced Security using Hybrid Encryption Algorithm".:International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 4, Issue 7, July 2016.

[2]. Jaimin Patel "Secure Hashing Algorithm and Advance Encryption Algorithm in Cloud Computing": International Journal of Computer and Information Engineering Vol:11, No:6, 2017.

[3].G. Tejaswini Bhorkar, "A Survey of Password Attacks and Safe Hashing Algorithms International Research Journal of Engineering and Technology (IRJET), Volume: 04 Issue: 12 Dec-2017

[4]. Chaitya B. Shah, Drashti R. Panchal "Secured Hash Algorithm-1: Review Paper" International Journal For Advance Research In Engineering And Technology, Volume 2, Issue X, Oct 2014

[5]. Ankit Kumar Jain, Rohit Jones, Puru Joshi, "Survey of Cryptographic Hashing Algorithms for Message Signing" IJCST Vol . 8, Issue 2, April - June 2017

[6]. K. Saravanan and A. Senthilkumar, "Theoretical Survey on Secure Hash Functions and issues", International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 2 Issue 10, October – 2013

[7]. Swathi S V, II Lahari P M, III Bindu A Thomas,"Encryption Algorithms: A Survey",International Journal of Advanced Research in Computer Science & Technology (IJARCST 2016),Vol. 4, Issue 2 (Apr. - Jun. 2016).

[8]. Muhammad Faheem Mushtaq, Sapiee Jamel, Abdulkadir Hassan Disina, Zahraddeen A. Pindar, at .el,"A Survey on the Cryptographic Encryption Algorithms",(IJACSA) International Journal of Advanced Computer Science and Applications,Vol.

[9]. 8, No. 11, 2017.