Enhancing the Data Security in Cloud by Implementing Hybrid (Rsa & Aes) Encryption Algorithm

Vishwanath S Mahalle
Department of Computer Science & Engineering
Shri Sant Gajanan Maharaj College of Engineering,
Shegaon, Maharashtra, India
vsmahalle@ssgmce.ac.in

Aniket K Shahade
Department of Information Technology
Shri Sant Gajanan Maharaj College of Engineering,
Shegaon, Maharashtra, India
aniket.shahade11@gmail.com

***Abstract*—This paper presents Hybrid (RSA & AES) encryption algorithm to safeguard data security in Cloud. Security being the most important factor in cloud computing has to be dealt with great precautions. This paper mainly focuses on the following key tasks:**

1. Secure Upload of data on cloud such that even the administrator is unaware of the contents.

2. Secure Download of data in such a way that the integrity of data is maintained.

3. Proper usage and sharing of the public, private and secret keys involved for encryption and decryption.

The use of a single key for both encryption and decryption is very prone to malicious attacks. But in hybrid algorithm, this problem is solved by the use of three separate keys each for encryption as well as decryption. Out of the three keys one is the public key, which is made available to all, the second one is the private key which lies only with the user. In this way, both the secure upload as well as secure download of the data is facilitated using the two respective keys. Also, the key generation technique used in this paper is unique in its own way. This has helped in avoiding any chances of repeated or redundant key.

***Keywords*— Cloud; RSA Algorithm; AES Algorithm; Key Generation; Private key; Public key; Secret key; Authentication; Encryption; Decryption**

I. INTRODUCTION

Security is one of the most difficult task to implement in cloud computing. The paper basically deals with the security issues that are experienced during the storage of data on the cloud. The cloud vendors generally store the client's data and information in cloud without following any security measures.

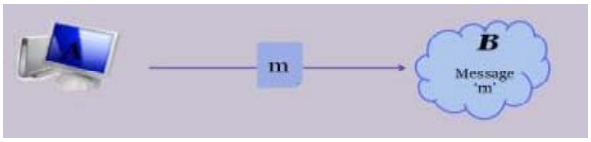


Fig.1. Data Uploaded in Cloud

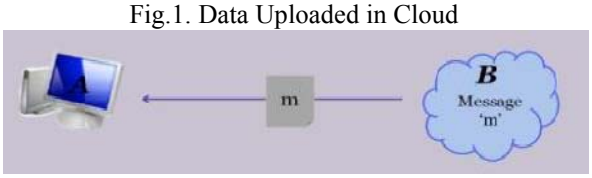


Fig.2. Data Downloaded from Cloud

Almost every cloud provider does not provide enough security measures to ensure the data safety and that's why clients waver keeping their data at some place which is very easy to be accessed by someone else.

Proposed System

The proposed system is implemented in Eye os that is one of the cloud providers. In order to apply security features, a hybrid encryption technique using the AES and RSA algorithms is used where 128 bit secret key for AES and 1024 bit key for RSA is used. Upload option leads to a generation of RSA public key-n, RSA public key-e, .RSA private key-d and AES secret key, user will require to save the RSA private key and AES secret key, As soon as user tries to upload the data on cloud, the data is first stored in a temporary directory and after calling AES and RSA algorithm, requiring the user to enter the AES secret key the file will get stored into the data base permanently corresponding to the user account and the temporary file get deleted.

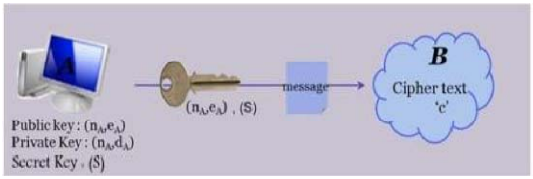


Fig.3. Data Uploaded in Cloud

Now, when the user wants to access the data stored in cloud or wants to download the data, it goes through the download procedure whereby user has to specify the filename to be downloaded and has to provide the AES secret and RSA private key which is kept secret by the user and is known only to him.

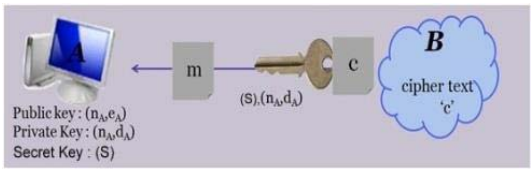


Fig.4. Data Downloaded from Cloud

II. MATERIALS AND METHODS

The proposed system basically consists of two modules
(I) Upload Module (II) Download Module

978-1-4799-7169-5/14/$31.00 ©2014 IEEE

(I)Upload Module: It consist of four parts

(i)Authentication: User authenticates himself to the Cloud with his unique username and the password.
(ii)Upload: This module allows user to upload his files in a secure way. Uploads the encrypted form of that data (file) in his document directory of cloud through this gateway.
(iii)Key Generation: A key generation is based on system timing.
(iv)Encryption: The data after uploading is first stored in the temporary file of the server that is in the Cloud. Then encrypt the data by using the public key of the user and stores the encrypted form of data in the documents of the user. The temporary files are then unlinked.

(II) Download Module: It consists of two parts

(i)Decryption: When user wants to download his secure data, he is prompted to enter his user name along with the secret private key. By using the private key of the user the cloud decrypts the data.
(ii)Download: Cloud send the Decrypted data to the user thereby giving the user his original data.

III. ALGORITHMS

(I)Upload Module:

Keygen procedure: It consist of different procedures

primeNoSelection():

This procedure is used to generate two distinct prime numbers P and Q of specified length (i.e. bigger than the size of integer) using current system time.

i. Set XSTRING = NULL
ii. Set YSTRING = NULL
iii. Repeat (iv) to (vi) until length (XSTRING) != specified length
iv. XSTRING = concat (XSTRING, getCurrentSystemTime);
v. YSTRIN = concat(YSTRING, etCurrentSystemTime*11);
vi. Wait(50)
vii. P=StringToBigInteger(XSTRING) //converts obtained string to big int
viii. Q=StringToBigInteger (YSTRING) //converts string to biginteger
ix. Repeat(xiv)until prime(P)!= TRUE
x. P = P + Biginteger("1") // building P a prime biginteger
xi. Repeat(xvi)untilprime(Q)!= TRUE
xii. Q = Q + Biginteger("1") // building Q a prime biginteger
xiii. DONE.

keyGenerator (P,Q)

This procedure is used to generate RSA public key, private key pair by using two big prime integers P and Q, E is Public key and D is Private Key.

i. N=P*Q //Calculating public key N
ii. PHI = (P – 1) * (Q – 1)
iii. Take E, a BigInteger value of length length(N)–1
iv. Repeat (v) until gcd(E,PHI) != 1
v. E=E+1 //selecting random number E which is relatively prime to PHI
vi. D=modInverse(E,PHI) // E *D =1 (mod PHI)
vii. Publish E as Public key and D as Private key
viii. Repeat (xiv) and (xv) until length (AES_key)!=specified length
ix. AES_key=concat(AES_key, Rand om no.); // Generating key
x. Wait(50)
xi. S = stringToBiginteger(AES_key) //converting number obtained to big integer
xii. Publish S as secret key
xiii. DONE

Uploadfile():

Procedure uploads desired file in the cloud by encrypting the contents of file and storing it in cloud.

i. START
ii. CALL keygen.class
iii. Retrieve username filename and secret key from user
iv. Check for validity of file
v. If file already exists on cloud
vi. Display ERROR_MESSAGE
vii. ELSE upload file to temporary directory
viii. CALL AESEncrypt(FILENAME, SECRET KEY) // Encrypting
ix. Open file in read mode, FD := open (FILENAME)
x. Open file in append mode, FD1 := open(concat(FILENAME,X))
xi. CALL Encryption(FD,FD1, USER NAME) // Encrypting with RSA
xii. close FD
xiii. close FD1
xiv. Move FD1 file from temporary directory to Documents. //Encrypted file saved
xv. Unlink FD file from temporary directory. // deletion of file from temp folder
xvi. END

AESEncrypt(FILENAME, SECRET KEY) :
This procedure is used to encrypt the given file with the help of FILENAME, SECRET KEY and AES encryption algorithm

i. START
ii. Retrieve filename and secret key
iii. Create AESEncrypt object
iv. CALL setkey(secret key)
v. CALL KeyExpansion()
vi. CALL ReadMessage()
vii. CALL test.Encrypt()
viii. addRoundkey
ix. CALL subByte()
x. CALL shiftRow()
xi. CALL mixcols()
xii. END

Encryption (FILENAME, USERNAME, PUBLIC KEY) //RSA Encryption:
This procedure encrypts the input file by taking username, filename from user and public key fstored in the database generated by the keygenerator

i. Read file in FD
ii. Retrieve PUBLIC_KEY_E and UBLIC_KEY_N from database with the help of USERNAME
iii. WHILE FD!=EOF
iv. FOR (i = 0 to i = 41) and (fd != EOF)
v. CALL Stringtoascii_conversion()
vi. Set ASCIISTRING:= concat(ASCIISTRING,OFFSET)
vii. END FOR
viii. NUM=stringToInteger(ASCIISTRING)
ix. SetENCRYPT:= [NUM$_{PUBLIC_KEY_E}$] mod[PUBLIC_KEY_N]
x. Set TEMPVAR := integerTo String [ENCRYPT]
xi. Write TEMPVAR to FD1
xii. Write a Delimiter '|' to FD1
xiii. DONE.

(II) Download Module:

download (): This procedure sends the decrypted file to user

i. RetrieveUSERNAME,FILENAME, PRIVATE_KEY from user
ii. Check validity of user entered data
iii. IF(validdata)THEN//RSA Decryption Starts
iv. Call decryption(FILENAME, USERNAME, PRIVATE_KEY)
v. ENDIF
vi. CALL AESdecrypt(FILENAME, SECRET KEY) //AES Decryption
vii. Setting proper header for downloading of decrypted file
viii. Unlink the decrypted file
ix. DONE.

Decryption (FILENAME,USERNA ME,PRIVATE_KEY) //RSA Decryption
This procedure is used to decrypt the given file with the help of RSA PRIVATE_KEY, PUBLIC_KEY_N and RSA decryption algorithm]

i. Retrieve PUBLIC_KEY_N from database using USERNAME
ii. Open file for reading,
FD1 = open(FILENAME)//Decryption with RSA
iii. Open another file for writing,
FD2 = open(substr(FILENAME,-1))
iv. Repeat (v)to (ix) until FD1 != EOF
v. Repeat (vi) and (vii) until CH = '|'
vi. NUM = concat(NUM, CH)
vii. Read a charcter from FD1 into CH
viii. SCIISTRING= (NUMPRIVATE_KEY) mod (PUBLIC_KEY_N)
ix. CallSTRS=asciiToString (ASCIISTRING)
x. Write STRS to FD2.
xi. Close FD1 and FD2
xii. CALLAESDecrypt(FILENAME, SECRETKEY);//Decryptionwith AES
xiii. DONE.

AESdecrypt (FILENAME,SECREKEY)
This procedure is used to decrypt the given file with the help of FILENAME, SECRET KEY and AES decryption algorithm

i. START
ii. Retrievefilenameand secret key
iii. Create AESdecrypt object
iv. CALL setkey(secret key)
v. CALL KeyExpansion()
vi. CALL ReadMessage()
vii. CALL test.decrypt()
viii. addRoundkey
ix. CALL subByte()
x. CALL shiftRow()
xi. CALL mixcols()
xii. END

asciiToString (ASCIISTRING)
This procedure is used to convert ASCII value to String which is further used in decrypting the file contents

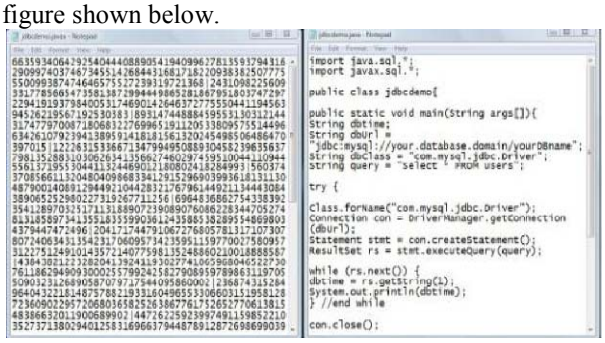
i. Declare STRS, OFFSET
ii. FOR(I=0;I< length (ASCIISTRING); I = I + 3)
iii. STR1 = "";
iv. STR2="";(TR1= substr (ASCIISTRING, I , 3)

v. STR1 = STR1 – OFFSET
vi. STR2 = char(STR1)
vii. STRS=concat(STRS, STR2)
viii. END FOR
ix. Return STRS.

IV. Results and Benefits of the Proposed System

1. High Security is provided by using hybrid (RSA&AES) encryption algorithm.
2. 1024 bit of RSA and 128 bit of AES keys are used, so one cannot guess the private key with the help of the public keys generated.
3. Using hybrid encryption leads to high security of text file and make the access of original file by intruders near to impossible.
4. If a user logins and forgets to log out or leaves the system idle. In that case if a trespasser tries to download the data from the system then that person will be asked to enter the private key. In any case if a trial to guess that private key and then try to download, he will get the data as seen on the left side of the diagram shown below. He not get the original data.

5. To download the data in a secure way the user is always required to enter the private key and secret key. Since the private key and secret is not even known to the Cloud's Administrator. Thus, the main advantage of proposed system is that even the Cloud's Administrator cannot access the data of the user. In case he tries to see the data he will see it in the encrypted form as seen on the left side of the figure shown below.



V. Conclusion

This paper proposed a hybrid encryption algorithm using RSA and AES algorithms for providing data security to the user in the Cloud. The biggest advantage it provides us is that the keys are generated on the basis of system time and so no intruder can even guess them there by giving us increased security along with convenience. Private Key and Secret key is only known to the user and therefore user's private data is not accessible to anyone not even the Cloud's Administrator. The main purpose behind using RSA and AES encryption algorithm is that it provides three keys i.e. public key for encryption, and private key and secret key for decryption. The data after uploading is stored in an encrypted form and can be only decrypted by the private key and the secret key of the user. The main advantage of this is that data is very secure on the cloud.

References

[1] Prof. Vishwanath S. Mahalle, "Implementing RSA encryption algorithm to enhance the data security of cloud in cloud computing", International journal of pure & applied research in engineering and technology, 2013, volume 1(8):220-227, ISSN-2319-507X IJPRET.

[2] Uma Somani, Kanika Lakhani, Manish Mundra "Implementing Digital Signature with RSA Encryption Algorithm to Enhance the Data Security of Cloud in Cloud Computing" 2010 1st International Conference on Parallel, Distributed and Grid Computing (PDGC) – 28-30 Oct, 2010 IEEE.

[3] (U.S.) Nicholas. Carr, fresh Yan Yu, "IT is no longer important: the Internet great change of the high ground - cloud computing," The Big Switch:Rewining the World, from Edison to Google, , CITIC Publishing House, October 2008.

[4] Ya-Qin Zhang, the future of computing in the "cloud-Client", The Economic Observer reported, http://www.sina.com.cn, 2008 Nian 07 Yue 12 Ri.