International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

# SECURE FILE STORAGE ON CLOUD USING HYBRID CRYPTOGRAPHY

P. Oliva Joicy[1], V. Prathyusha[2], M. Rekha[3]

Dr. Swapna[4]

Department of Computer Science and Engineering, Stanley College of Engineering and Technology for Women, Telangana, India

**Abstract.** The proposed model is liable to meet the required security needs of data centred of cloud. AES, DES and RSA are used for the encryption of file slices takes minimum time and has maximum throughput for encryption and decryption from other symmetric algorithms.

The idea of splitting and merging adds on to meet the principle of data security. The hybrid approach when deployed in cloud environment makes the remote server more secure and thus, helps the cloud providers to fetch more trust of their users. For data security and privacy protection issues, the fundamental challenge of separation of sensitive data and access control is fulfilled.

Cryptography technique translates original data into unreadable form. Cryptography technique is divided into symmetric key cryptography and public key cryptography. This technique uses keys for translate data into unreadable form. So only authorized person can access data from cloud server. Cipher text data is visible for all people.

**Keywords:** Cryptography, Encryption, Decryption, Steganography, Cipher text, Advanced encryption standard, Data encryption standard, Plain text.

## 1. Introduction

### 1.1 About Project

Today's technologies are growing at very fast speed and deliver the user with many attractive services to reduce the burden of large volumetric data storage and maintenance. Nowadays, many online services are applicable which provides all services and data online such as e-messaging, billing, e-transaction, e-mail etc. All these services required user's data online for processing. This data may be any confidential information, which is required by user to be safe from any malicious activity like-healthcare information, bank transaction, credit card details, etc. A high requirement arises for security and protection of data from any unauthorized user as leakage of confidential information may result in serious impairment to user. This increases the security requirement of confidential data before actually migrating it over online internet access. We need to develop a sound, safe and secure framework to protect our confidential data from any such malicious attack.

There is a need to convert confidential data into some another form, which becomes inexplicable for any attacker and only authorized users are able to understand that exactly what data is communicated. One of the major techniques to achieve this requirement is cryptography, also known as “code making” or “code generation”.

It allows us to translate a message into unreadable form for malicious attacker. Another part of this approach is known as cryptanalysis and “code breaking.

To maintain security requirements such as data confidentiality and its integrity, authentication is a prime concern to prevent any unauthorized user from sniffing the data to be communicated between two or more parties. To overcome this issue owner needs to first encrypt it before actually transferring it over cloud service provider and provide only authorized users with decryption key, as shown in figure 1. Introduction of cryptography mechanism protects the user data and ensures that the confidential information of user is protected and secure from any unauthorized user access and malicious attack. Unauthorized user tries to hamper the user’s data by altering or modifying it. But due to lack of key it become a tedious task for attacker. Only authorized user has the authority and capability to revert back the converted received data into original form. An efficiency issue arises when users are revoked and private key encryption will not work in that case. Cloud data security issues can be resolved by answering some question as-

- Identification of users for accessing owner confidential data.
- Who has the authority to modify the existing data within cloud?
- Will the authorized user use the same private key or a unique key is assigned to each individual user?
- Where exactly the keys will be stored or how to transmit security key(s) to all authorized user?
- Which techniques are used to encrypt and decrypt the cloud data?

## 1.2 Objectives of the Project

The proposed model is liable to meet the required security needs of data center of cloud. AES, DES and RSA are used for the encryption of file slices takes minimum time and has maximum throughput for encryption and decryption from other symmetric algorithms.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

The idea of splitting and merging adds on to meet the principle of data security. The hybrid approach when deployed in cloud environment makes the remote server more secure and thus, helps the cloud providers to fetch more trust of their users. For data security and privacy protection issues, the fundamental challenge of separation of sensitive data and access control is fulfilled. Cryptography technique translates original data into unreadable form. Cryptography technique is divided into symmetric key cryptography and public key cryptography. This technique uses keys for translate data into unreadable form. So only authorized person can access data from cloud server. Cipher text data is visible for all people.

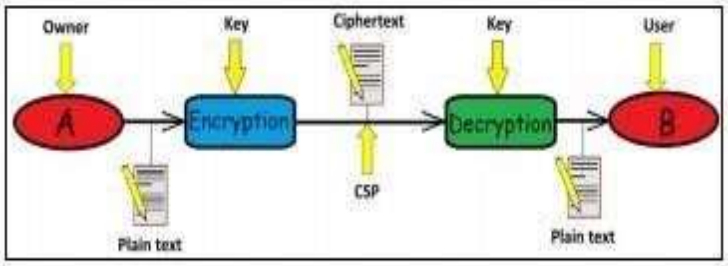



Fig. 1. Data flow in Cloud Computing

### 1.3 Scope of the Project

Data security is one of the most researched topic. When user shares personal data to third party applications data security methods should be efficient.

Though there are existing algorithms for encryption there are problems in those methods. Hybrid cryptography can be used in any platform for improving data security.

## *2. Literature Survey*

### 2.1 Existing System

- In existing methods only single algorithms are mostly used for providing security for data.
- AES, DES, RSA are mostly used algorithms.
- ECC algorithm was also used for encryption for mobile based applications.
- Existing system is a manual & time taking process.

## 2.2 Proposed System

In order to provide security for data in cloud there are many types of techniques which are already proposed like AES, DES, RSA but in existing methods most of the time only single type of encryption was used either AES, OR DES, OR RSA based on user requirement but in this system main problem is each encryption is done using encryption keys if these keys are exposed in any case entire data is lost so we need effective method.

## *3. Proposed Architecture*

The purpose of the design phase is to arrange an answer of the matter such as by the necessity document. This part is that the opening moves in moving the matter domain to the answer domain. The design phase satisfies the requirements of the system. The design of a system is probably the foremost crucial issue warm heartedness the standard of the software package.

It's a serious impact on the later part, notably testing and maintenance. The output of this part is that the style of the document. This document is analogous to a blueprint of answer and is employed later throughout implementation, testing and maintenance. The design activity is commonly divided into 2 separate phases System Design and Detailed Design.

System Design conjointly referred to as top-ranking style aims to spot the modules that ought to be within the system, the specifications of those modules, and the way them move with one another to supply the specified results.

At the top of the system style all the main knowledge structures, file formats, output formats, and also the major modules within the system and their specifications square measure set. System design is that the method or art of process the design, components, modules, interfaces, and knowledge for a system to satisfy such as needs. Users will read it because the application of systems theory to development.

Detailed Design, the inner logic of every of the modules laid out in system design is determined. Throughout this part, the small print of the info of a module square measure sometimes laid out in a high level style description language that is freelance of the target language within which the software package can eventually be enforced.

In system design the main target is on distinguishing the modules, whereas throughout careful style the main target is on planning the logic for every of the modules.

Start is the process where owner will login to application and upload data using file upload feature then uploaded data is split in to three parts equally and then first part is encrypted using AES, DES, RSA and data is stored in cloud and database.

When client want to download data decryption is performed using decryption phase and get final result.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

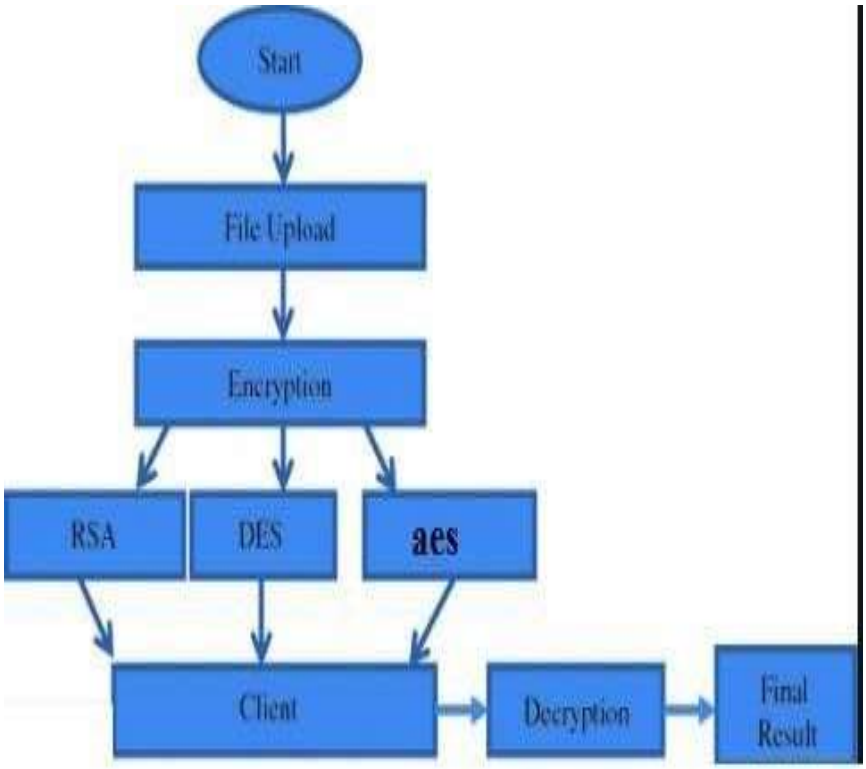



Figure: Architecture diagram

## *4. Implementation*

### 4.1 Algorithm

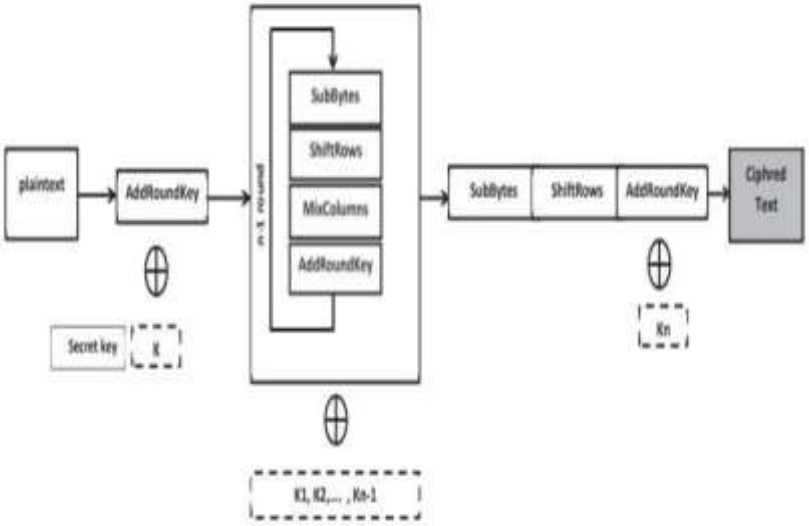
**AES**

The AES algorithm is easy to implement, it consumes less memory and it uses keys of 128, 192 or 256 bits. The required number of rounds (i.e., linear and non-linear transformations), depend on the key size

AES has four steps such as Byte sub, shift row, mixed column and add round key. The only nonlinear step which is responsible to create confusion in the data is byte sub, the remaining steps are nonlinear.

In other words we can say that in shift row, mixed column and add round key we are only applying permutation operation for the sake to diffusion.

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org




```
Cipher(byte in[4 * Nb], byte out[4 * Nb], word w[Nb * (Nr + 1)])
begin
   byte  state[4,Nb]

   state =  in

   AddRoundKey(state, w)                          // See Sec. 5.1.4

   for round = 1 step 1 to Nr – 1
      SubBytes(state)                             // See Sec. 5.1.1
      ShiftRows(state)                            // See Sec. 5.1.2
      MixColumns(state)                           // See Sec. 5.1.3
      AddRoundKey(state, w + round * Nb)
   end for

   SubBytes(state)
   ShiftRows(state)
   AddRoundKey(state, w + Nr * Nb)

   out = state
```

**RSA**

In RSA we uses the same algorithm for encryption and decryption, we need pair of keys, public and private key we have three principal steps

**Algorithm 1:** RSA

1. Begin.
2. $m$ = the ASCII code of the plaintext.
3. $c$ = the ASCII code of the ciphertext.
4. Choose two large prime numbers $p$ and $q$ (+100 digits).
5. Compute $\varphi(n) = (p - 1)(q - 1)$.
6. Compute $n = pq$
7. Choose any number $1 < e < \varphi(n)$ that is coprime to $\varphi(n)$.
8. Compute the value of $d$ such that $(d * e) \bmod \varphi(n) = 1$.
9. Public key is $(e, n)$.
10. Private key is $(d, n)$.
11. The encryption of $m$ is $c = m^e \bmod n$.
12. The decryption of $c$ is $m = c^d \bmod n$.
13. End.

**DES**

DES algorithm consumes least encryption time and AES algorithm use least memory usage, Encryption time differs in case of AES algorithm and DES algorithm.

```
Cipher (plainBlock[64], RoundKeys[16, 48], cipherBlock[64])
{
    permute (64, 64, plainBlock, inBlock, InitialPermutationTable)
    split (64, 32, inBlock, leftBlock, rightBlock)
    for (round = 1 to 16)
    {
        mixer (leftBlock, rightBlock, RoundKeys[round])
        if (round!=16) swapper (leftBlock, rightBlock)
    }
    combine (32, 64, leftBlock, rightBlock, outBlock)
    permute (64, 64, outBlock, cipherBlock, FinalPermutationTable)
}
```

### 4.2 Code Implementation

```
Import os
Import datetime
Import hashlib

from flask import Flask, session, url_for, redirect, render template, request, abort, flash from database import
db_connect,user_reg,owner_reg,owner_login,upload_file,owner_viewfiles,upload_
clouddata,user
_request,owner_request,user_lastdownload from database import
onwer_viewdata,user_loginact,user_viewfile,user_viewfiledata,user_down,verify_u
ser,verify_user
2,user_finaldown,owner_update
from werkzeug.utils import secure_filename
app = Flask(__name__) app.secret_key = os.urandom(24) @app.route("/") def FUN_root(): return
render_template("index.html")
@app.route("/owner") def FUN_admin():
 return render_template("owner.html")
@app.route("/ownerlogact",methods = ['GET','POST']) def owner_logact():
if request.method == 'POST':
status=owner_login(request.form['username'],request.form['password'])
```

```
 if status == True:session['username'] = request.form['username']
return render_template("ownerhome.html",m1="sucess")
else:
return render_template("owner.html",m1="Login Failed")
@app.route("/user/") def FUN_student():      #clg_names = get_clgnames()
 return render_template("user.html")
@app.route("/userreg/") def FUN_userreg():   return render_template("userreg.html")
@app.route("/userregact", methods = ['GET','POST']) def user_regact():
if request.method == 'POST':

 status =
user_reg(request.form['username'],request.form['password'],request.form['dob'],request                .form['emai
l'],request.form['city'],request.form['contactno'])
if status == True: return render_template("userhome.html",m1="Login sucess")       else:
 return render_template("user.html",m1="Login failed")
@app.route("/userlogact",methods = ['GET','POST']) def user_logact():
if request.method == 'POST':
status=user_loginact(request.form['email'],request.form['password'])
if status == True:          session['email'] = request.form['email']
return render_template("userhome.html",m1="sucess")
else:
return render_template("user.html",m1="Login Failed")
@app.route("/userhome") def user_home():
 return render_template("userhome.html")
  @app.route("/vf/") def user_vf():
  viewfile = user_viewfile(session['email'])
return render_template("vf.html", viewfiledata = viewfile)
@app.route("/vf1/", methods = ['GET', 'POST']) def user_vf1():
fname = request.args.get('filename')         owner = request.args.get('owner')
data = request.args.get('data')         print(fname,owner,data,session['email'])
 check = user_viewfiledata(fname,owner,data,session['email'])
 if check == True:
return render_template("vf.html",m1="Request_sent_to_CloudB")
else:
 return render_template("vf.html",m1="Request_failed")
 @app.route("/download/") def user_download():
downloaddata = user_down(session['email'])
 return render_template("download.html", downloads = downloaddata)
```

```
@app.route("/downloadact/") def user_downloadact():
downloaddata = user_down(session['email'])
return render_template("downloadact.html", downloadview = downloaddata)

@app.route("/ownerreg/") def FUN_ownerreg():
return render_template("ownerreg.html")
@app.route("/ownerregact", methods = ['GET','POST']) def FUN_ownerregact():
if request.method == 'POST':
status =
owner_reg(request.form['username'],request.form['password'],request.form['dob'],request .form['email'],r
equest.form['city'],request.form['contactno'])
if status == True:
    return render_template("ownerhome.html",m1="Login sucess")
else:
    return render_template("owner.html",m1="Login failed")
@app.route("/ownerhome") def FUN_ownerhome():
  return render_template("ownerhome.html")
@app.route("/Upload", methods = ['GET','POST']) def owner_upload():
 if request.method == 'POST':  file = request.files['inputfile']
check = upload_file(request.form['fname'],file,session['username'],"No","No")
if check == True:
    return render_template("fileupload.html",m1="success")
else:
    return render_template("fileupload.html",m1="Failed")
@app.route("/fileupload")  def FUN_fileupload():
    return render_template("fileupload.html")
@app.route("/ownerviewfiles") def FUN_ownerviewfiles():
viewdata = owner_viewfiles(session['username'])  notes_table = zip([x[0] for x in viewdata],\
                [x[1] for x in viewdata],\
                [x[2] for x in viewdata],\
                [x[3] for x in viewdata],\
                [x[4] for x in viewdata])
return render_template("ownerviewfiles.html",showdata = notes_table)

@app.route("/viewencfiles")  def  owner_viewencfiles():
splitdata  = onwer_viewdata(session['username'])
```

```
return render_template("viewencfiles.html",spliinfo = splitdata)
@app.route("/logout") def FUN_logout():
return render_template("index.html")
@app.route("/response", methods = ['GET','POST']) def owner_response():
  fname1 = request.args.get('filename')
data1 = request.args.get('data')
owner1 = request.args.get('owner')
email1 = request.args.get('email')
print("uuuuuuuuuuuuuuuuuuuuuuuuuuuudfdfdfdfdfdfduuuuuuuuuuuuuuuuuuuuu") result =
owner_request(fname1,owner1,email1)
rdata = result
status = owner_update(rdata,fname1,owner1,email1)
 if status == True:
return render_template("vuserreq.html",m1="sucess")
@app.route("/verify1", methods = ['GET','POST']) def user_verfiy():
 filename =request.form['filename']
dkey = request.form['dkey']
 vdat=verify_user(filename,dkey)
 if vdat: return render_template("download2act.html",vdatavsend = vdat)
else:   return render_template("download.html",m1="no data in table")
@app.route("/verify2", methods = ['GET','POST']) def user_verfiy2():
filename =request.form['filename']
key = request.form['dkey']     dat=verify_user2(filename,key)
    if dat:
      return render_template("download3act.html",datavsend = dat)

    else:
      return render_template("download.html",m1="no data in table")
```

## *5. Result*

Home

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org


Fig 5.1 Output Screen -1 Owner login


Fig 5.2 Output Screen-2

Owner Registration

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org


Fig 5.3 Output Screen-3

Owner Home


Fig 5.4 Output Screen-4

File upload

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org


Fig 5.5 Output Screen-5


File upload

Fig 5.6 Output Screen-6

View split data

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

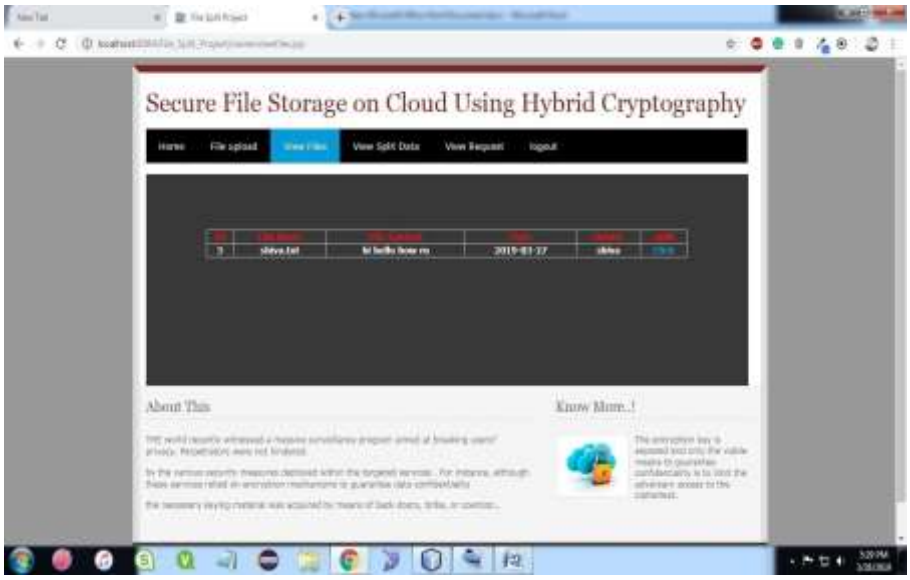



Fig 5.7 Output Screen-7

View Files


Fig 5.8 Output Screen-8

Decrypt File part 1

International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org


Fig 5.9 Output Screen-9

Decrypt File part 2


Fig 5.10 Output Screen-10

## *6. Conclusion*

In the proposed method the strength of AES-DES-RSA hybrid increase the level of security as compared to the existing technique where only DES is used. In this method the message is encrypted with an AES DES RSA and keys of cipher text is hidden inside an image using LSBimage steganography.

Steganography, specially shared with the cryptography is a stronger tool which allows exchanging information secretly. With the fast growth of digital technology and internet, steganography has highly developed a lot in a past few years. It will test the knowledge of the attacker about both cryptography and steganography.

If an attacker is able to extract data from image then he has to crack the hybrid cryptography then only he will get the exact data. A result of proposed technique shows that the encryption time is better than the existing technique.

It provides a more security compared to the existing one. Brute force attack on this technique is very difficult to use as there is use of AES RSA for DES key. In future other steganography techniques may be used with hybrid cryptography for more security.

## *7. Future Scope*

It is not possible to develop a system that makes all the requirements of the user. User requirements keep changing as the system is being used. Sone of the future enhancements that can be done to this System are:

- As the technology emerges, it is possible to upgrade the system and can be adaptable to desired environment.
- Based on the future security issues, security can be improved using emerging technologies like single sign-on.

In future, more advanced symmetric or asymmetric cryptographic algorithm can be implemented to ensure more data security from any malicious activity. Apart from this, some access control techniques can also be included to perform the access control and authenticate the user before data transmission from CSP to user.

## *8. References*

1. [1] Sinkov A., “Elementary Cryptanalysis – A Mathematical Approach”, Mathematical Association of America, 1996
2. [2] L. Arockiam, S. Monikandan “Data Security and Privacy in Cloud Storage using Hybrid Symmetirc Encryption Algorithm”, International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, Issue 8, pp 3064-3070, August 2013.
3. [3] http://www.nku.edu (Fall 2006 Chris Christensen)
4. [4] G.L.Prakash, M.Prateek and I.Singh, ‘Data Encryption and Decryption Algorithms using Key Rotations for Data Security in Cloud System’, International Journal of Engineering and Computer Science, Vol. 3, Issue 4, April 2014, pp. 5215-5223.
5. [5] Juels, Ari and Burton S.Kaliski Jr. ‘PROs: Proofs of retrievability for Large Files’, Procedding of the 14th conference on Computer and communication security, ACM 2007.

6. [6] Fadhil Salman Abed, “A Proposed Method of Information hiding based on Hybrid Cryptography and Seganography”, International Journal of Application or Innovation in Engineering & Management, Vol. 2, Issue 4, April 2013.
7. [7] Dai Yuefa, Wu Bo, et al. "Data security model for cloud computing."Proceedings of the 2009 International Workshop on Information Security and Application (IWISA 2009) Qingdao, China.., pp 141-144, 2009.
8. [8] Frank Gens et al., “Cloud Computing 2010 An IDC Update “http://www.cionet.com/data/files/groups/cloud%20computing%20210 %20-%20an%20idc%20update.pdf,2010.
9. [9] D.Zissis and D.Lekkas, ‘Addressing cloud computing security issues’, Elsevier Journal of Future Generation Computer Systems, Vol. 28,pp 583-592..
10. [10] Amit joshi and Bhavesh Joshi “A Randomized Approach for Cryptography” International Conference on Emerging Trends in Network and Computer Communications (ETNCC), pp. 293- 296, April 2011.