



**SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

**27 OCTOBER 2018**

# **CARD PAYMENT SECURITY USING RSA**

## **A Project Report**

**Under the Guidance of,**

**Prof. Dr.MARIMUTHU K**

**By**

**17BCI0011 – HARSHIT DADHICH**

**17BCI0024 – ACHAL GOYAL**

**17BCI0058 – SAURABH AMBER**

## **DECLARATION BY THE CANDIDATE**

I/We hereby declare that the project report entitled “**CARD PAYMENT SECURITY USING RSA**” submitted by us to Vellore Institute of Technology, Vellore in partial fulfilment of the requirement for the award of the degree of **B. Tech (Program Name CSE)** is a record of J- component of project work carried out by us under the guidance of **Prof. Dr.MARIMUTHU K.** We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore Institute of Technology, Vellore.

Date : 27-10-2018

Signature of the faculty

Signature of the Candidate

# TABLE OF CONTENTS

## **1. Introduction**

- 1.1 Abstract
- 1.2 Background

## **2. Overview and Planning**

- 2.1 Proposed Work
- 2.2 Hardware Requirements
- 2.3 Software Requirements

## **3. Literature Survey and Review**

- 3.1 Literature Summary

## **4. Methodology**

- 4.1 Method Used
- 4.2 Applications

## **5. System Implementation**

- 5.1 Code
- 5.2 Results and discussion

## **6. Conclusion**

- 6.1 Conclusion
- 6.2 Future Work

## **7. References**

## **ABSTRACT**

RSA algorithm is asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. Public Key and Private Key. As the name describes that the Public Key is given to everyone and Private key is kept private. This project proposed an implementation of a complete and practical RSA encrypt/decrypt solution on CARD PAYMENT SECURITY, the study of RSA public key algorithm. In addition, the encrypt procedure and code implementation is provided in details.

## **INTRODUCTION**

Due to increasing e-commerce activity nowadays, there is a need for some encryption technique to ensure security and a way to ensure that the user's data are securely stored in the database. Thus the system introduces RSA for this purpose. The RSA algorithm is a kind of asymmetric encryption algorithm which appeared in 1978. The algorithm is public key encryption algorithm which is a widely accepted and implemented by public. The use of RSA in this the system makes the process more secure. Now the bank transactions can be done securely without worrying about attacker getting access to the database as the data will be in encrypted form.

RSA is the first algorithm known to be suitable for signing as well as encryption, and one of the first great advances in public key cryptography. It is named for the three MIT mathematicians who developed it — Ronald Rivest, Adi Shamir, and Leonard Adleman.

RSA is very widely used today for secure Internet communication (browsers, S/MIME, SSL, S/WAN, PGP, and Microsoft Outlook), operating systems (Sun, Microsoft, Apple, Novell) and hardware (cell phones, ATM machines, wireless Ethernet cards, Mondex smart cards, Palm Pilots). Prasithsangaree and his colleague Krishnamurthy have analyzed the Energy Consumption of RC4 (RSA) and AES Algorithms in Wireless LANs in the year 2003. They have evaluated the performance of RC4 and AES encryption algorithms in [9]. The performance metrics were encryption throughput, CPU work load, energy cost and key size variation. Experiments show that the RC4 is fast and energy efficient for encrypting large packets. However, AES was more efficient than RC4 for a smaller packet size.

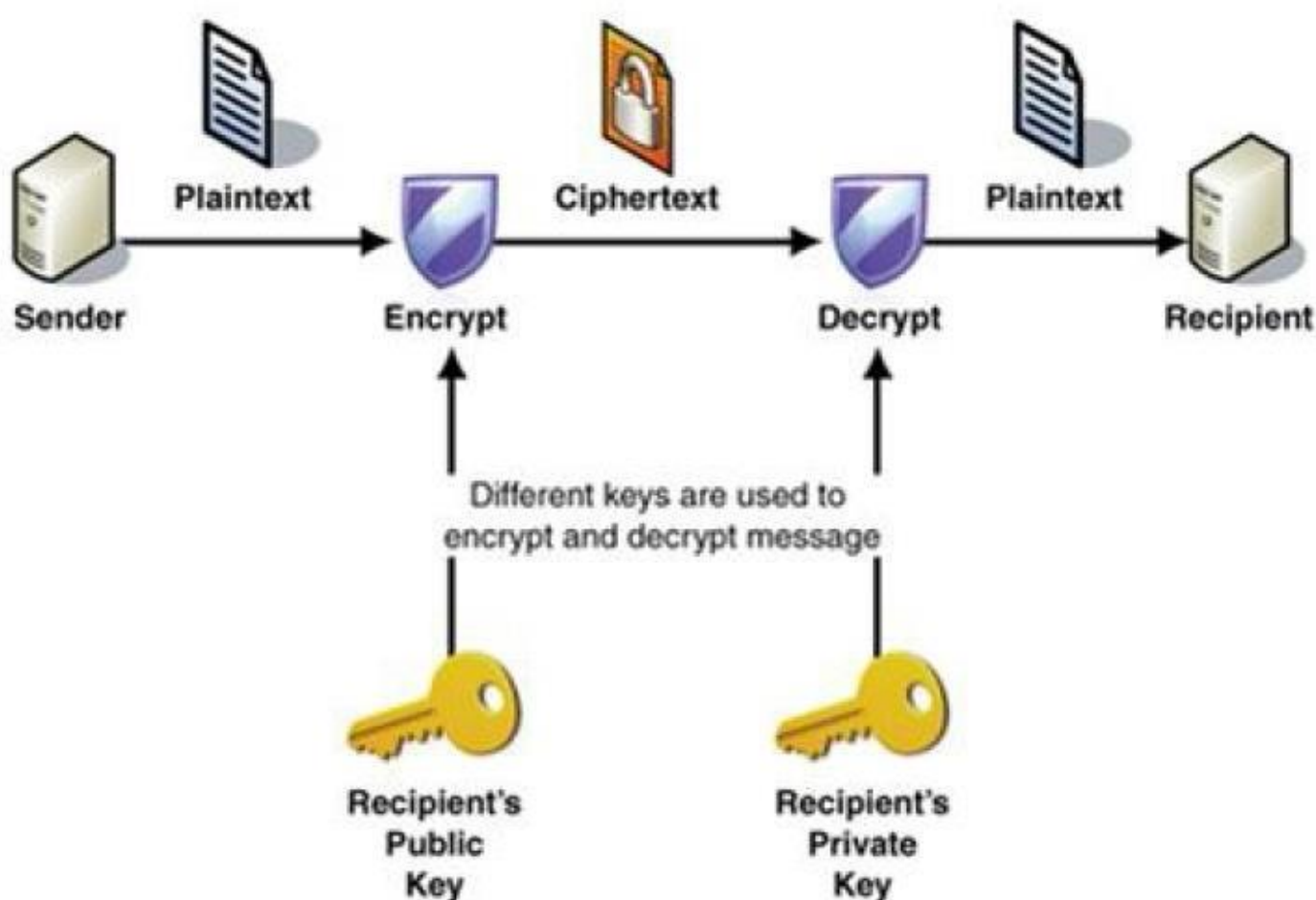


Fig. 2: cryptographic encryption of plain text

The design of the RSA security software partly evolved from the need for an all embracing information security system and partly from the need for a user friendly package that can fulfil any large ecommerce organization's information security needs.

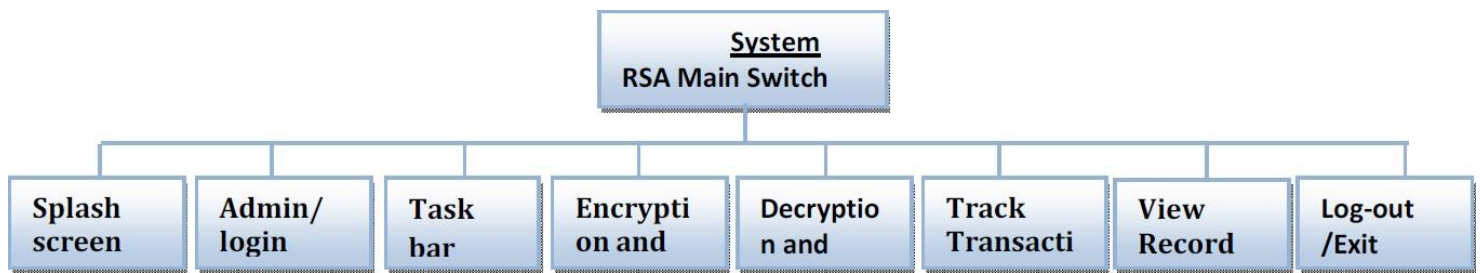
## **LITERATURE SURVEY**

Sistu Sudheer Ku mar, A. Srinivas Reddy (May -2015) [1] Authentication is a critical part of any trustworthy computing system which ensures that only authorized individuals can log on to the system. Here, ATM Security has always been one of the most prominent issues. ATM machines generally authenticates by using ATM card and PIN number to perform transactions. This paper discusses design of ATM system that will improve the authentication of customer while using ATM. Here is possible scenario that an individual's ATM card falling into wrong hands by knowing PIN number and forget ATM card is difficult to perform ATM transaction. So to clear all these problems we are implementing this system using "One Time Password (OTP)" and "Personal Identification Number (PIN)" combination in order to improve authentication of customer using ATM machine to perform transaction without having any ATM cards.

Ezeofor C. J, Ulasi A. G. (December 2014) [2], this paper presents an analysis of network data encryption and decryption techniques used in communication systems. In network communication systems, exchange of information mostly occurs on networked computers, mobile phones and other internet based electronic gadgets. Unsecured data that travels through different networks are open to many types of attack and can be read, altered or forged by anyone who has access to that data. To prevent such an attack, data encryption and decryption technique is employed

Nentawe Y. Goshwe (July 2013) [4], one of the principal challenges of resource sharing on data communication network is its security. This is premised on the fact that once there is connectivity between computers sharing some resources, the issue of data security becomes critical. This paper presents a design of data encryption and decryption in a network environment using RSA algorithm with a specific message block size. The algorithm allows a message sender to generate a public key to encrypt the message and the receiver is sent with a generated private key using a secured database. An incorrect private key will still decrypt the encrypted message but to a form different from the original message.

# DESIGN



A Modular is a system component that provides services to other components but would not normally be considered as a separate system as in. A separable component is one that is interchangeable with others for assembling into units of differing size, complexity or function as in. Therefore RSA cryptosystem is designed along modular techniques.

This necessitated the decomposition of the system into clearly defined subsystems such that the initial requirements specifications were met. The software system comprises the following subsystems: splash-screen subsystem, Admin/login subsystem, Task bar/Key generation subsystem, Encryption subsystem, Decryption subsystem, Track Transaction subsystem, View record subsystem, Log out/Exit subsystem.

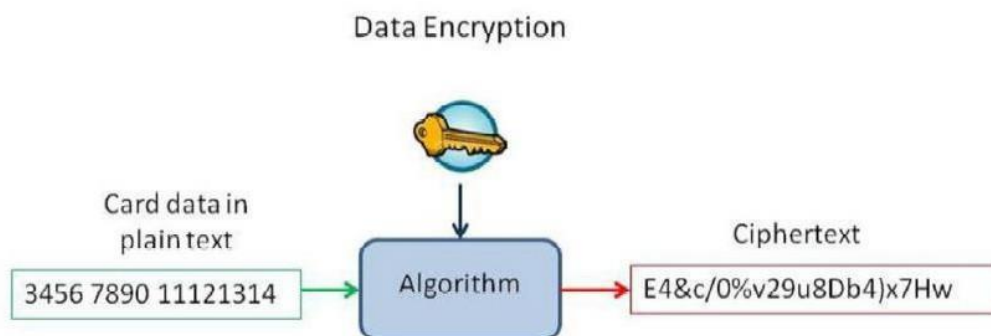


Fig. 4: Symmetric Data encryption

The public key can be freely distributed without the key management challenges of symmetric keys since it can only encrypt and never decrypt data.

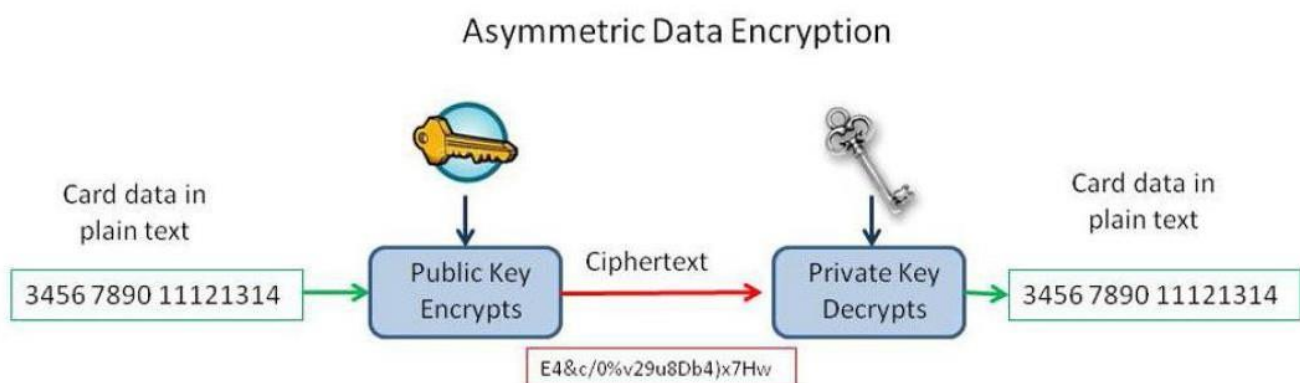


Fig. 5: Asymmetric Data encryption

In a payment environment, the public key can be distributed to a merchant or to the end POS device, and that device can store the key in hardware or software. Even if that key is extracted by someone who shouldn't have rights to it, all that the person can do is encrypt data with the key; he can't decrypt anything. On the other hand, the corresponding private key where the decryption occurs must be handled very securely.

The RSA algorithm is the most commonly used public key encryption algorithm in asymmetric cryptography.

Two keys are used: Public Key and Private Key.

**Key generation:** RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

Choose two distinct prime number  $p$  and  $q$ .

For security purposes, the integers  $p$  and  $q$  should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.

Compute  $n = pq$ .

$n$  is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.

Compute  $\phi(n) = \phi(p)\phi(q) = (p-1)(q-1) = n - (p+q-1)$ , where  $\phi$  is Euler's totient function.

Choose an integer  $e$  such that  $1 < e < \phi(n)$  and  $\gcd(e, \phi(n)) = 1$ ; i.e.,  $e$  and  $\phi(n)$  are coprime.

$e$  is released as the public key exponent.

$e$  having a short bit-length and small Hamming weight results in more efficient encryption – most commonly  $2^{16} + 1 = 65,537$ . However, much smaller values of  $e$  (such as 3) have been shown to be less secure in some settings.

Determine  $d$  as  $d \equiv e^{-1} \pmod{\phi(n)}$ ; i.e.,  $d$  is the multiplicative inverse of  $e$  (modulo  $\phi(n)$ ).

This is more clearly stated as: solve for  $d$  given  $d \cdot e \equiv 1 \pmod{\phi(n)}$

This is often computed using the extended Euclidean algorithm. Using the pseudocode in the Modular integers section, inputs  $a$  and  $n$  correspond to  $e$  and  $\phi(n)$ , respectively.

$d$  is kept as the private key exponent.



The public key consists of the modulus  $n$  and the public (or encryption) exponent  $e$ . The private key consists of the modulus  $n$  and the private (or decryption) exponent  $d$ , which must be kept secret.  $p$ ,  $q$ , and  $\phi(n)$  must also be kept secret because they can be used to calculate  $d$  as in

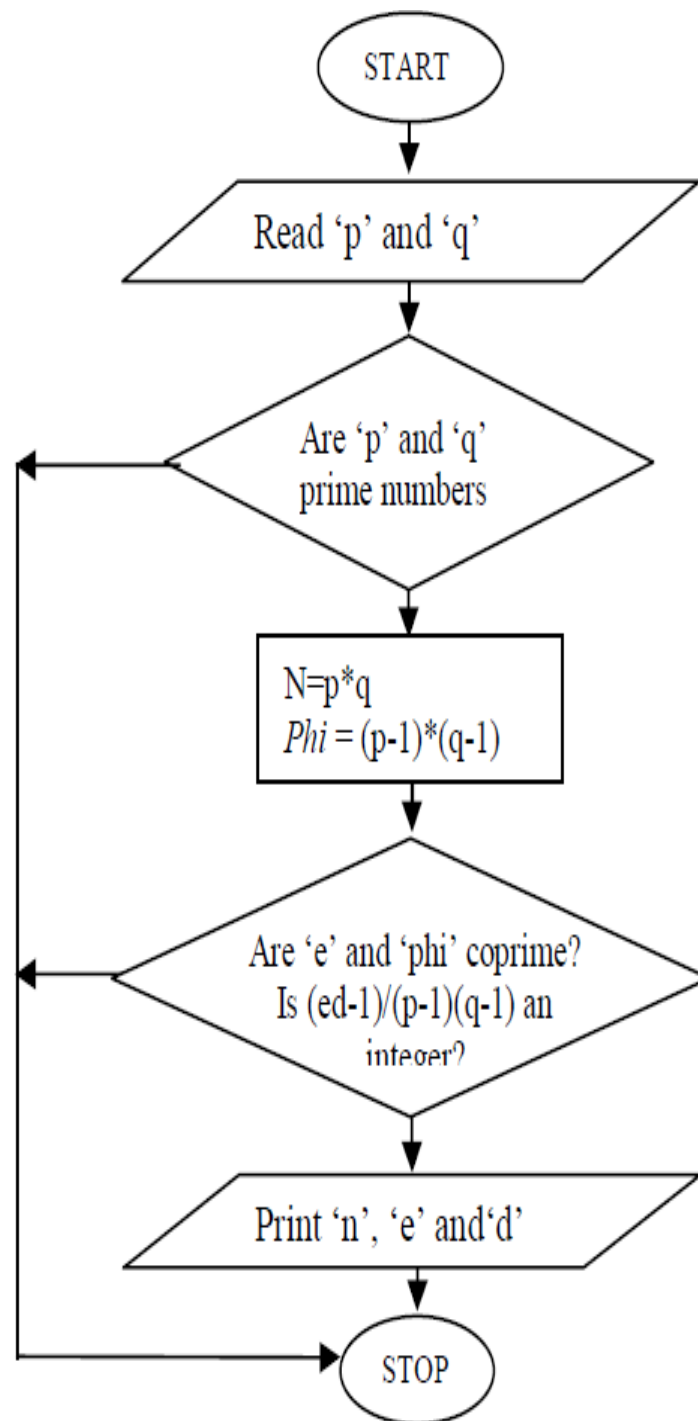


Fig. 6: flow chart illustrating the RSA Key generation

After getting the public and private key the main thing is how to encrypt and decrypt using RSA.

**RSA Encryption:** Alice transmits her public key (n, e) to Bob and keeps the private key d secret. Bob then wishes to send message M to Alice. He first turns M into an integer m, such that  $0 \leq m < n$  by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to

$$c \equiv m^e \pmod{n}$$

This can be done quickly using the method of exponentiation by squaring. Bob then transmits c to Alice.

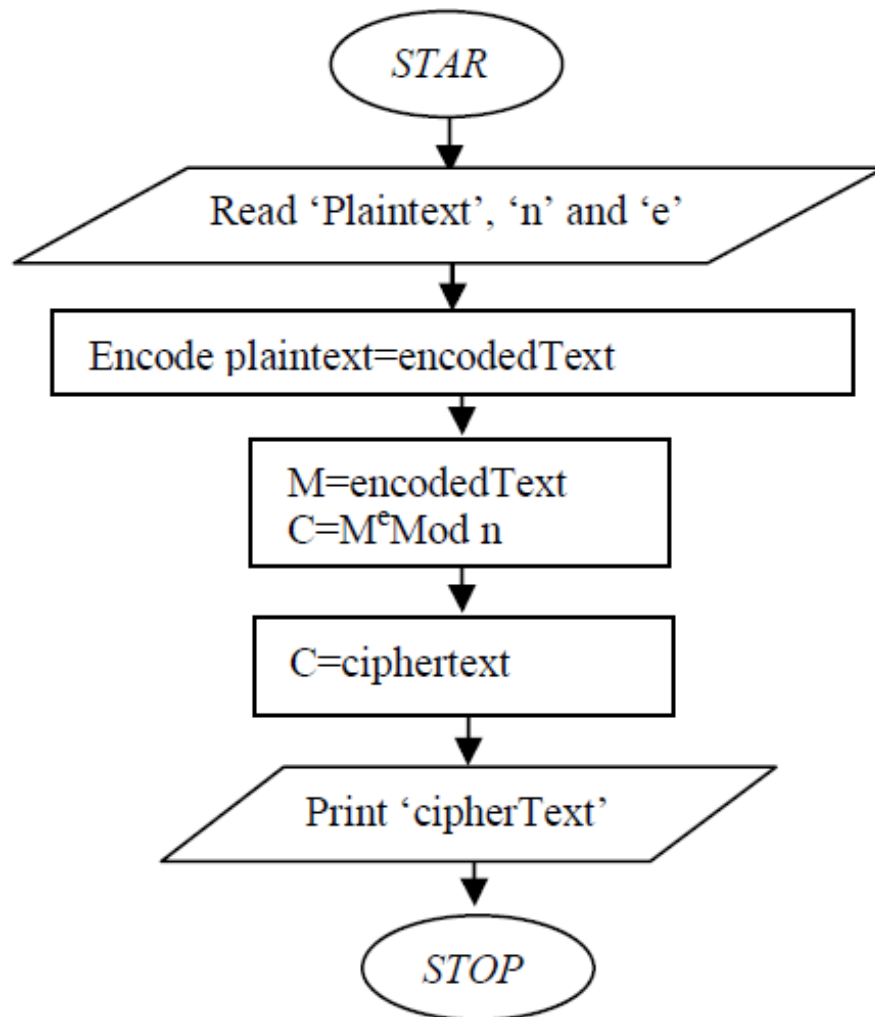


Fig.7: A flow chart illustrating the RSA Encryption Algorithm

**RSA Decryption:** Alice can recover  $m$  from  $c$  by using her private key exponent  $d$  via computing

$$m \equiv c^d \pmod{n}$$

Given  $m$ , she can recover the original message  $M$  by reversing the padding scheme.

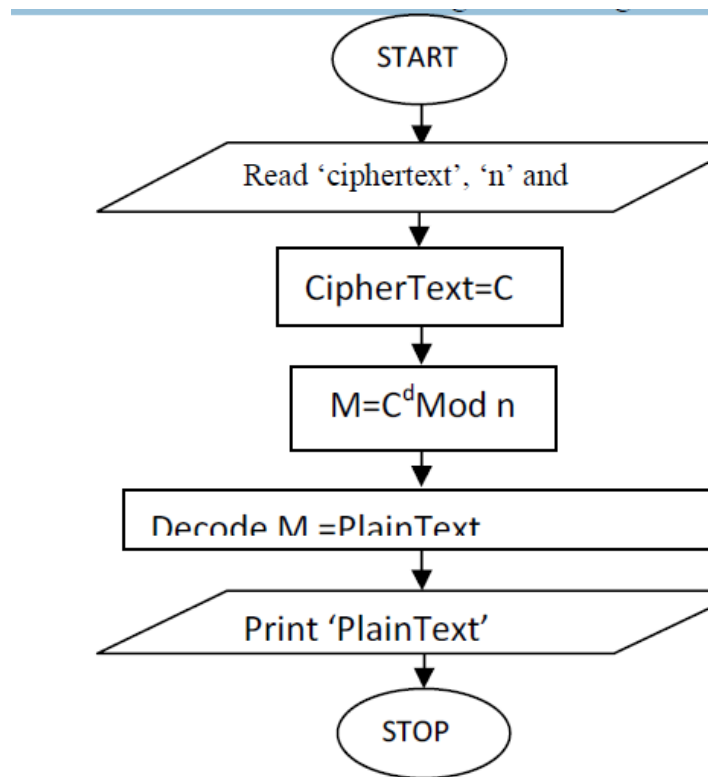


Fig. 8: Flow Chart illustrating the RSA Decryption Algorithm

## **Modules**

**Register:** User will have to register in order to get access to the system.

**Login:** User will have to provide his username and password in order to login to the system.

**Product Listing:** Here all the products can be viewed by the user along with its other details like short description, cost, and also image of the product.

**Payment:** In order to perform any transaction here the user needs to provide his bank information like his Account no, Card no, CVV no and PIN no to make the payment.

**Encryption:** The PIN, Account No, CVV no are encrypted using RSA

## **Advantages**

RSA is very high speed of encryption .

Your sensitive details like your bank details are now secured using RSA.

Now the bank transactions can be done securely without worrying about attacker getting access to the database as the data will be in encrypted form.

## **Application**

This system can be used in any online activity which requires users to perform payment online.

## **Code:**

```
//#include<iostream>
```

```
//#include<string.h>
```

```
# include <stdio.h>
```

```
//# include <fstream>
```

```
# include <string.h>
```

```
# include <iostream>
```

```
# include <stdlib.h>
```

```
# include <ctype.h>
```

```
# include <conio.h>
```

```
using namespace std;
```

```
struct user
```

```
{
```

```
_____string user_id;
```

```
_____string user_name;
```

```
int user_ccn;  
string user_address;  
int user_balance;  
string user_password;  
struct user *next;  
}*first=NULL;
```

```
void enqueue();  
//void updation();  
void display();  
void user_account();
```

```
int gcd(int a,int b)  
{  
return b==0 ? a:gcd(b,a%b);  
}
```

```
int check_banker_id(string id)  
{  
string check_id;  
int count=0;  
cin.ignore();  
cout<<"ENTER BANKER ID: ";  
getline(cin,check_id);  
if(id.length()==check_id.length())  
{  
for(int i=0;i<id.length();i++)  
{  
if(id[i]==check_id[i])  
count++;  
else  
break;  
}  
}
```

```

        if(count==id.length())
            return 1;
        else
            return 0;
    }
    else
        return 0;
}

```

```

int check_banker_password(string password)
{
    string check_password;
    cout<<"Enter 10 digit password: ";
    char a;
    for(int i=0;i<10;i++)
    {
        a=getch();
        check_password=check_password+a;
        cout<<"*";
    }
    int count=0;
    if(password.length()==check_password.length())
    {
        for(int i=0;i<check_password.length();i++)
        {
            if(password[i]==check_password[i])
                count++;
            else
                break;
        }
        if(count==check_password.length())
            return 1;
    }
}

```

```

        else
            return 0;
    }
    else
        return 0;

}

int main()
{

    string banker_id="2345";
    char password[11]="incredible";
    char a;
    string check_password;
    int check;//if k==1 then true else false
    int select;
    while(1)
    {
        cout<<"\nselect 1 for banker , 2 for user, 3 for exit: ";
        cin>>select;
        if(select==1)
        {
            check=check_banker_id(banker_id);
            if(check==1)
            {
                check=check_banker_password(password);
                if(check==1)
                {
                    system("cls");
                    cout<<"\t\t\t.....ADMINISTRATOR
PAGE.....";
                    cout<<"\n\n";
                    while(1){

```

```
cout<<"\nChoose from the following option:\n1. add  
user\n2. update user\n3.view user details\n4. Main Page\n\nENTER THE OPTION: ";
```

```
cin>>select;  
switch(select)  
{  
    case 1:  
    {  
        enqueue();  
        break;  
    }  
    /*case 2:  
    {  
        updation();  
        break;  
    }*/  
    case 3:  
    {  
        display();  
        break;  
    }  
    case 4:  
    {  
        main();  
    }  
    default:  
    {  
        cout<<"\nINVALID  
INPUT!!!!!!\nTRY AGAIN";  
    }  
}  
}  
}  
}  
else  
{
```



```

        cout<<"\nWronng Password.....\n";
    }
}
else
{
    cout<<"\nWrong Banker id.....\n";
}
}
else if(select==2)
{
    user_account();
}
else
    exit(0);
}
return 0;
}

```

```

void enqueue()
{
    int p,q,n,toitent,c,msg,e,z,k,i,d;
    cout<<"enter two large prime numbers(p and q): ";
    cin>>p>>q;
    n=p*q;
    toitent=(p-1)*(q-1);
    cout<<"select the value of e: ";
    for(i=1;i<toitent;i++)
    {
        z=gcd(i+1,toitent);
        if(z==1)
            cout<<i+1<<" ";
    }
}

```

```

    cout<<" ";
    cin>>e;

    string id,name;
    string address,password;
    int ccn,balance=0;

    system("cls");

    cout<<".....ADD
USERS....." <<endl;

    cout<<"Enter the following details":
    cout<<"\nUser Name: ";

    cin.ignore();
    getline(cin,name);
    cout<<"User id: ";

    //cin.ignore();
    getline(cin,id);
    cout<<"Enter ccn: ";

    cin>>ccn;
    cout<<"Enter Address: ";

    cin.ignore();
    getline(cin,address);
    cout<<"Create new password(5): ";

    //cin.ignore();
    //getline(cin,password);
    cin>>password;
    cout<<"enter the initial balance: ";

    cin>>balance;

    struct user *temp;

    //temp=(struct user *)malloc(sizeof(struct user));

//    int msg;

    msg=ccn;

    for(int i=0;i<e;i++)

    {

        k=k*msg;

```

```

    k=k%n;
}

//cout<<"\nencrypted ccn: "<<ccn;
cout<<"\nencrypted ccn is : "<<k<<endl;

temp=new(struct user);
temp->user_id=id;
temp->user_name=name;
temp->user_address=address;
temp->user_ccn=ccn;
temp->user_balance=balance;
temp->user_password=password;
temp->next=NULL;
if(first==NULL)
    first=temp;
else
{
    struct user *s;
    //s=(struct user*)malloc(sizeof(struct user));
    s=new(struct user);
    s=first;
    while(s->next!=NULL)
        s=s->next;
    s->next=temp;
}
}

void display()
{
    struct user *temp;
    temp=(struct user*)malloc(sizeof(struct user));
    if(first==NULL)
        cout<<"\nNo records available\n";

```

```

else
{
    int num;

    cout<<"press 1 to see all the records, 2 for specific record: ";

    cin>>num;

    if(num==1)
    {
        temp=first;
        while(temp!=NULL)
        {
            cout<<"\nuser id: "<<temp->user_id;

            cout<<"\nuser_name: "<<temp->user_name;

            cout<<"\nccn: "<<temp->user_ccn;

            cout<<"\naddress: "<<temp->user_address;

            cout<<"\nBalance: "<<temp->user_balance;

            cout<<"\nPassword: "<<temp->user_password;

            cout<<"\n\n\n";

            temp=temp->next;
        }
    }

    else if(num==2)
    {
        int flag=0;

        string want_id;

        cout<<"\nEnter the id: ";

        cin.ignore();

        getline(cin,want_id);

        temp=first;

        while(temp!=NULL)
        {
            if(temp->user_id==want_id)
            {
                cout<<"\nRECORD FOUND!!!";
            }
        }
    }
}

```

```

        cout<<"\nuser id: "<<temp->user_id;

        cout<<"\nuser_name: "<<temp->user_name;

        cout<<"\nccn: "<<temp->user_ccn;

        cout<<"\naddress: "<<temp->user_address;

        cout<<"\nBalance: "<<temp->user_balance;

        cout<<"\nPassword: "<<temp->user_password;

        cout<<endl;

        flag=1;

        break;

    }

    temp=temp->next;

}

if(flag==0)

    cout<<"\nRECORD NOT FOUND\n";

}

else

{

    cout<<"Wrong Input, TRY AGAIN!!!";

    display();

}

}

}

```

```

int check_user_id()

{

    string id;

    struct user *temp;

    temp=new(struct user);

    cout<<"Enter the user id : ";

    cin.ignore();

    getline(cin,id);

```

```

        int flag=0;

    }

void user_account()
{
    if(first==NULL)
        cout<<"\nNo user record!!!"<<endl;
    else
    {
        int flag=0;
        string id;
        struct user *temp;
        temp=new(struct user);
        cout<<"\nEnter the user id : ";
        //cin.ignore();
        //getline(cin,id);
        cin>>id;
        temp=first;
        while(temp!=NULL)
        {
            if(temp->user_id==id)
            {
                flag=1;
                break;
            }
            else
                temp=temp->next;
        }
        if(flag==0)
            cout<<"\nUser id NOT FOUND\n";
        else
        {

```

```

flag=0;

string password;

cout<<"\nenter the 5 digit password: ";

/*cin.ignore();

char a;

for(int i=0;i<5;i++)

{

    a=getch();

    password=password+a;

    //cout<<"*";

}*/

cin>>password;

while(temp!=NULL)

{

    if(temp->user_password==password)

    {

        flag=1;

        break;

    }

    temp=temp->next;

}

if(flag==0)

    cout<<"\nInvalid password ";

else

{

    cout<<"user found";

    cout<<"\nuser id : "<<temp->user_id;

    cout<<"\nuser name: "<<temp->user_name;

    cout<<"\nuser ccn: "<<temp->user_ccn;

    cout<<"\nuser address: "<<temp->user_address;

    int num;

    cout<<"\npress 1 for payment process, 2 for main menu, any number for exit ";

    cin>>num;

```

```

if(num==1)
{
    string id;

    cout<<"\nenter the credit card number";

    cin>>id;

    //getline(cin,id);

    struct user *target;

    target=new(struct user);

    target=first;

    flag=0;

    while(target!=NULL)
    {

        if(target->user_id==id)

        {

            flag=1;

            break;

        }

        target=target->next;

    }

    if(flag==0)
    {

        cout<<"\nINVALID credit card NUMBER.";

    }

```

else

```

{

    int amount;

    cout<<"Enter the amount want to pay: ";

    cin>>amount;

    if(amount > temp->user_balance)

        cout<<"\nInsufficient balance";

    else

    {

```



```
        temp->user_balance-=amount;
        target->user_balance+=amount;
    }

}

}

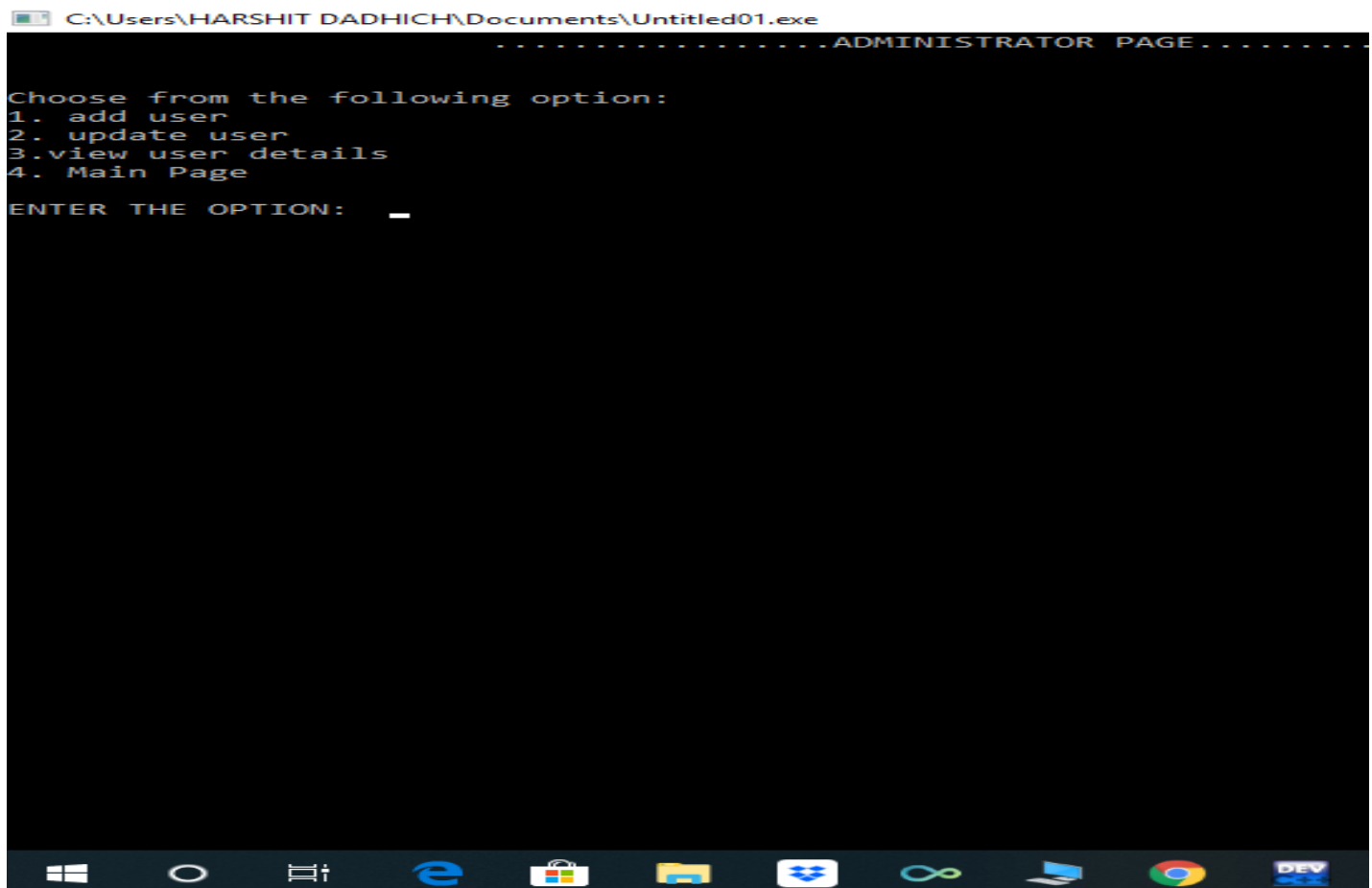
}

}

}

}
```

**Output:**



```
C:\Users\HARSHIT DADHICH\Documents\Untitled01.exe
.....ADMINISTRATOR PAGE.....

Choose from the following option:
1. add user
2. update user
3.view user details
4. Main Page

ENTER THE OPTION: _
```

Select C:\Users\HARSHIT DADHICH\Documents\Untitled01.exe

```
.....AD
Enter the following details
User Name: harsh
User id: 1234
Enter ccn: 123567890
Enter Address: rajasthan
Create new password(5): harsh
enter the initial balance: 20000000

encrypted ccn is : 487

Choose from the following option:
1. add user
2. update user
3.view user details
4. Main Page

ENTER THE OPTION:
```

|



```
select 1 for banker , 2 for user, 3 for exit: 2  
Enter the user id : 1234  
enter the 5 digit password: harsh  
user found  
user id : 1234  
user name: harsh  
user ccn: 123567890  
user address: rajasthan  
press 1 for payment process, 2 for main menu, any number for exit
```

```
press 1 for payment process, 2 for main menu, any number for exit 1  
enter the credit card number1234  
Enter the amount want to pay: 400
```

## **REFERENCES**

[1]Sistu Sudheer Kumar, A. Srinivas Reddy , May-2015, “ A survey on theft

prevention during ATM transaction without ATM cards”, eISSN: 2319-1163 | pISSN: 2321-7308 Volume: 04 Special Issue: 06 | NCEITCS-2015 | May-2015.

[2]Ezeofor C. J, Ulasi A. G., December 2014, “Analysis of Network Data Encryption & Decryption Techniques in Communication

Systems ” International Journal of Innovative Research in Science Engineering and Technology (An ISO 3297: 2007 Certified

Organization) Vol. 3, Issue 12, December 2014 ISSN: 2319 -8753.

[3]Sivakumar T, Gajjala Askok, k. Sai Venuprathap, August 2013, “ Design and Implementation of Security Based ATM theft Monitoring system”, International Journal of Engineering Inventions e-ISSN: 2278-7461, p-ISSN: 2319-6491 Volume 3, Issue 1 (August 2013) PP: 01-07.

[4]Nentawe Y. Goshwe , July 2013, “ Data Encryption and Decryption Using RSA Algorithm in a Network Environment” IJCSNS

[5]Boneh, Dan (1999). "Twenty Years of attacks on the RSA Cryptosystem". Notices of the American Mathematical Society. 46 (2): 203–213.

"Further Attacks ON Server-Aided RSA Cryptosystems", James McKee and Richard Pinch,  
<http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.33.1333>

A Course in Number Theory and Cryptography, Graduate Texts in Math. No. 114, Springer-Verlag, New York, 1987. Neal Koblitz, Second edition, 1994. p. 94

"common factors in  $(p - 1)$  and  $(q - 1)$ ",Viktor Dukhovni, openssl-dev Digest, Vol 9, Issue 4, <https://www.mail-archive.com/openssl-dev%40openssl.org/msg39736.html>, <https://www.mail-archive.com/openssl-dev%40openssl.org/msg39725.html>

[6] Johnson, J.; Kaliski, B. (February 2003). "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1". [www.ietf.org](http://www.ietf.org). Network Working Group. Retrieved 9 March 2016.