PROJECT TITLE: OPTIMIZED ELGAMAL ENCRYPTION ENHANCE SECURITY

A PROJECT REPORT

Submitted By,

MUNILAKSHMI GJ – 20BCI0190

INDEX PAGE NO

ABSTRACT

In the modern era, security is the biggest issue for all. Now a days, everyone using the internet for many reasons like transfer their money, data and etc. So, to improve the security of the current system we implement the cryptosystem which uses the Discrete Logarithm problem(DLP) for encryption that makes the encryption algorithm more secure. That is called Elgamal encryption. It is a public key cryptosystem. It uses asymmetric key for encryption and decryption. Existing elgamal cryptosystem uses only one key for encryption. These days there are such a large number of situations where unapproved clients access to imperative information.

In this project, we propose the modification of the ElGamal Cryptosystem to provide an additional level of security in the system. This modification allows the user to use multiple private keys to encrypt the message. And existing algorithm will convert the text into integer values which leads to 2*n size of the file. We have optimized the file size by converting the int values to corresponded character.

INTRODUCTION

Cryptography is an art of hiding a message. In cryptography, encrypt on is the process of encoding a message to human unreadable form. Confidentiality of the data will be achieved via the encryption. Encryption can be classified into symmetric and asymmetric encryption. Now a days so many encryption techniques are available. Elgamal encryption is on of that technique. Elgamal cryptosystem is a public key cryptosystem. It uses asymmetric key to encrypt and decryption data. This technique is based on the Diffie-Helman key exchange. It was first described by Taher Elgamal in the year of 1985. The advantage of usingthis technique is each time while we encrypt the message, it will gives different cipher text for same message. Existing elgamal encryption consists of three parts. They are key generation, encryption and decryption.

The security of the ElGamal Cryptosystem is based on the success of the Discrete Logarithm Problem (DLP). As long as the DLP remain hard, ElGamal will remain efficient. The security of the DLP is dependent on the size of the Prime modulo and the private keys used by the sender and the receiver. In the case of the Prime Modulo, the longer it is, the DLPbecomes more complicated, and the implementation of the ElGamal Cryptosystem would be more secure. When these keys are kept in secret, the ciphertext is going to be very hard to break, and when there are severalmany private keys, the level of security proportionally increases also.

Even though Existing algorithm uses the logic of discrete logarithmic problem, it uses only one key and if we choose small key it is easy for the attacker to crack the key. Also, it has to encrypt for every element for duplicate also. And ciphertext is twice as long as plain text. Also, it takes more execution time for encryption compare to other techniques. So, it is leads possible to some attacks like brute for attack,known plaintext attack on elgamal cryptosystem is to solve the discrete logarithm problem.

When we comes into applications the most common application of elgamal cryptosystem is Pretty Good Privacy(PGP)which is hybrid cryptosystem that uses both symmetric and asymmetric techniques. Otherapplications are internet voting, hybrid cryptosystem, part of digitalsignature and free GNU privacy guard.

Features added in it:

- **Using more than one key to encrypt the message**
- **Using a random number of keys to encrypt the message every time**
- **Generating new keys after every communication**
- **Padding message with random characters to hide length of message**
- **Mapping the message to Unicode characters (optimize the file size)**

Existing method

Existing elgamal encryption consists of three parts. They are key generation, encryption and decryption.

1. Bob generates public and private key:

- **Bob chooses a very large prime number p.**
- **From the p, he finds g which is the primitive root of p.**
- **Then he computes y = gx mod p.**
- **Bob publishes p, g and y as his public key and retains x as private key.**

2. Alice encrypts data using Bob's public key:

- **Alice selects a random integer k < p.**
- **Then she computes c1 = gk mod p.**
- **She multiples yk with M that is consider as c2.**
- **Then she sends (c1,c2).**

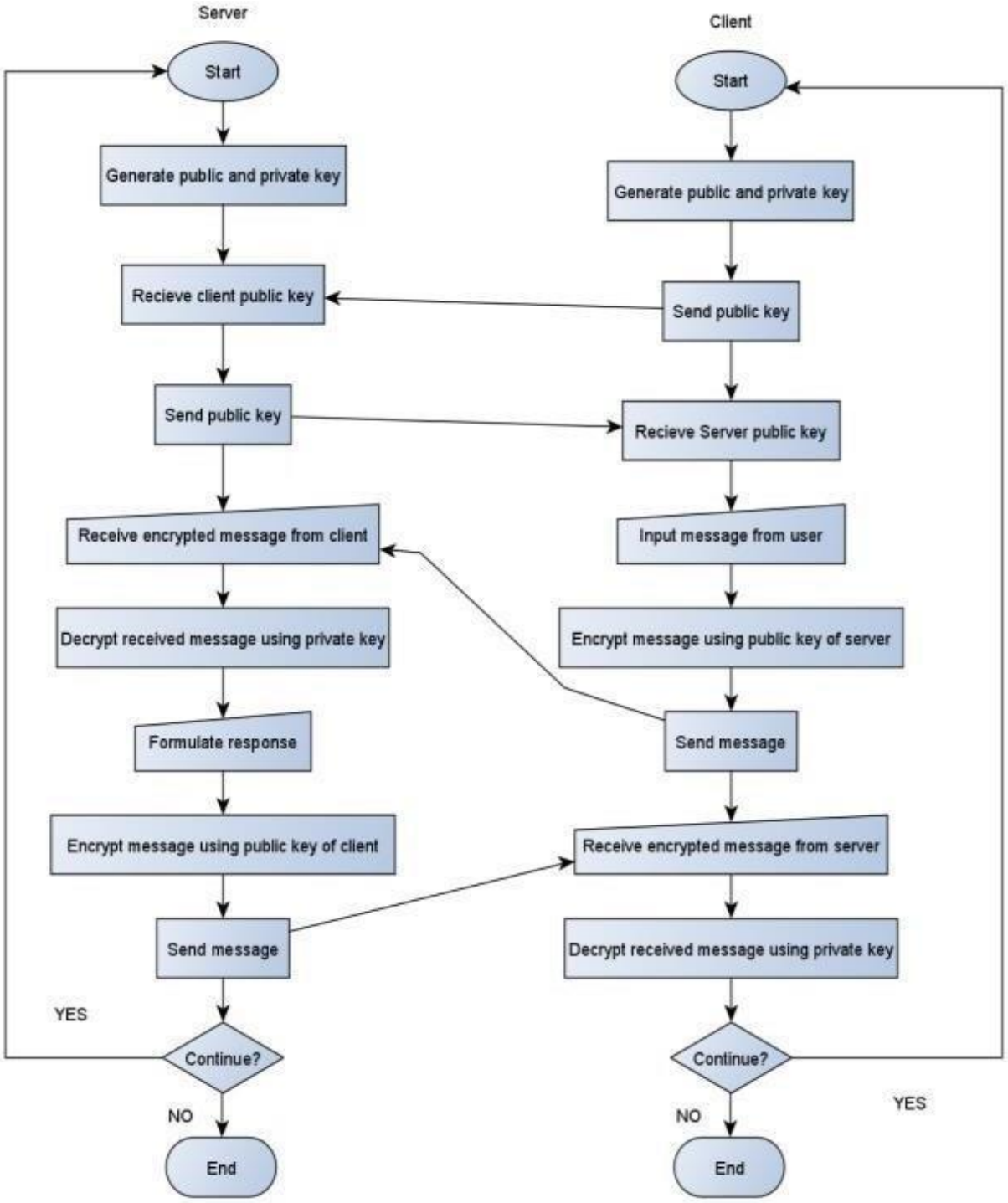
3. Bob decrypts the message:

- **Bob calculates s' = (c1x)-1.**
- **Then he multiplies c2 by s' to obtain M.**

PROPOSED SYSTEM

Existing algorithm uses discrete logarithmic problem it is very hard to crack the key using brute force attack. With modified and dedicated hardware we can crack the key. But our goal is to develop an enhanced Elgamal encryption system which system is not crack easily. Existing encryption method is using integer as a cipher text. But, in our proposed system we use character string as a cipher text so it reduce the file size. In this proposed algorithm it is impossible to break it via brute-force attack. And also Cipher text attack is not possible since attacker has no idea about the keys and length of the message.

FLOWCHART OF PROPOSED SOLUTION



Module wise description

Figure shows the how our proposed system works. Herewe explain it detailed.

1. Key generation (Server):

- **Choose a random prime number (p) and choose a random primitive root(g) of the prime number.**
- **Choose a random integer (n) as the number of keys (rounds) in our private key**
- **Generate n random integers (xn) which will act as our private key,applyyn = gx mod p to each of the integers to generate Y (list of integers).**
- **Send p, g, Y to client which acts as our public key.**

2. Encryption (Client):

- **Choose n length(Y) random numbers (kn)**
- **Compute Yk mod p using all numbers in Y and a and multiply them and store in one variable c and then compute c=c mod p.**
- **Compute a list A = gk mod p using all integers in a kn**

- **Pad the message with c random characters in the beginning and c/2 in the end.**
- **Encrypt message by multiplying c with message and then**
- **Convert them to characters resulting in encrypted message B**
- **Send A,B to Server**

3. Decryption (Server):

- **Receive A,B from client**
- **Compute c by applying Ax mod p on all the integers in A corresponds to x then multiply them together and mod them with p.**
- **Begin processing message from position c as the characters before it are just padding and end the decryption at c/2.**
- **Divide the Unicode value of each character in the message by cand then convert them back to a character.**
- **This is our decrypted message.**

Implementation Details and Analysis

We are implementing this using python language.

CODE

Elgamal.py

```
import math,random
import string

primel=[]
for i in range(76342,652423):
        f=1
        for j in range(2,int(math.sqrt(i))+1):
                if i%j == 0:
                        f=0
                        break
        if f:
                primel.append(i)

def primitive_root(p):
        if p == 2:
                return 1
        p1 = 2
        p2 = (p-1) // p1
        while(1):
                g = random.randint( 2, (p-1) )
                if not (pow( g, (p-1)//p1, p ) == 1):
                        if not pow( g, (p-1)//p2, p ) == 1:
```

```
                return g

def genkey():
    p=primel[random.randint(0,len(primel)-1)]
    g = primitive_root(p)
    n = random.randint(4,9)
    b=[]
    B=[]
    while len(b)!=n:
        x = random.randint(2,p-2)
        if x not in b:
            b.append(x)
    for i in range(n):
        B.append(pow(g,b[i],p))
    return [p,g,B,b]

def encrypt(z,n,p,g,B):
    a=[]
    while len(a)!=n:
        x=random.randint(2,p-2)
        if x not in a:
            a.append(x)
    c=1
    A=[]
    for i in range(n):
        sec = pow(B[i],a[i],p)
        A.append( pow(g,a[i],p))
        c*=sec
    c%=p
```

```
        if(c==1):
                c=5
        while(c>255):
                c=c//2
        mes=random.choices(string.ascii_letters + string.digits, k = c)
        for i in z:
                mes.append(i)
        mes+=random.choices(string.ascii_letters + string.digits, k = c//2)
        for i in range(0,len(mes)):
                w=(c*ord(mes[i]))
                mes[i]=chr(w)
        return [A,''.join(mes)]


def decrypt(n,A,b,p,l):
        s=[];l2=[]
        for i in range(n):
                l2.append(pow(A[i],b[i],p))
        c=1
        #B.append(pow(g,b[i],p))
        for i in l2:
                c*=i
        c%=p
        if(c==1):
                c=5
        while(c>255):
                c=c//2
        s=""
        ll = len(l)-c//2
```

```
        for i in range(c,ll):
                w=(ord(l[i])//c)
                s+=chr(w)
        return s
```

Client.py

```
import socket
import string
import random
import math as m
import pickle
from elgamal import *
cs = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = socket.gethostname()
port = 12345
cs.connect((host,port))
while True:
        k = genkey()
        recv = cs.recv(10000)
        serverkey = pickle.loads(recv)
        key=pickle.dumps(k[0:3])
        cs.send(key)
        # Key exchange done
        print("\nEnter message:")
        mes = input()
        x = encrypt(mes,len(serverkey[2]),serverkey[0],serverkey[1],serverkey[2])
```

```
        encmes=pickle.dumps(x)
        cs.send(encmes)

        recv = cs.recv(10000)
        encmes = pickle.loads(recv)
        print("\nEncrypted message: ",encmes[1])
        decmes = decrypt(len(k[3]),encmes[0],k[3],k[0],encmes[1])
        print("\nDecrypted message: ",decmes)
```

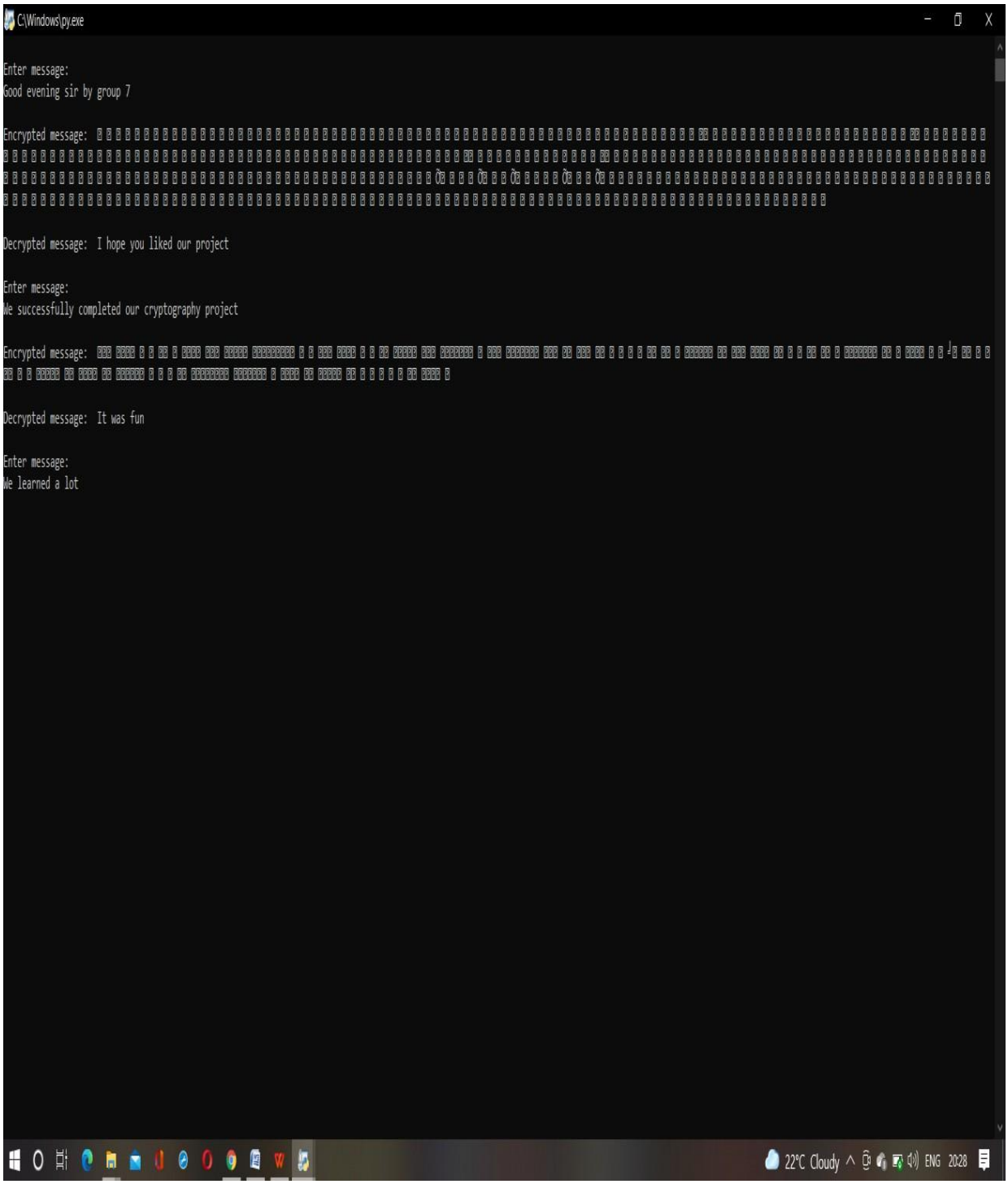
Server.py

```
import socket
import string
import random
import math as m
import pickle
from elgamal import *
ss = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
host = socket.gethostname()
port = 12345
ss.bind((host,port))
ss.listen(1)
c,addr = ss.accept()
while True:
        k = genkey()
        key=pickle.dumps(k[0:3])
        c.send(key)
        recv = c.recv(10000)
        clientkey = pickle.loads(recv)
        #key exchange done
        recv = c.recv(10000)
```

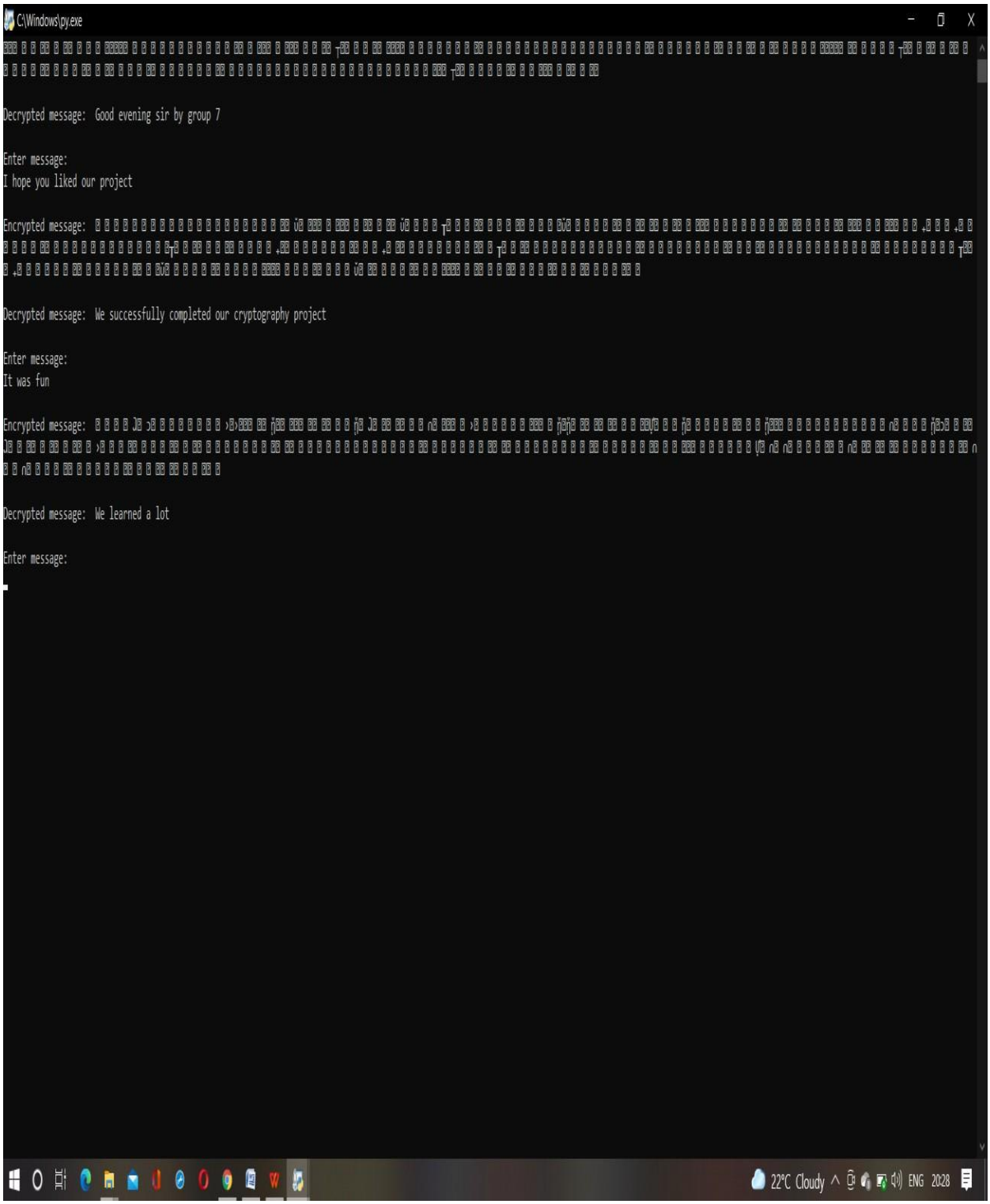
```
encmes = pickle.loads(recv)
print("\nEncrypted message: ",encmes[1])
decmes = decrypt(len(k[2]),encmes[0],k[3],k[0],encmes[1])
print("\nDecrypted message: ",decmes)

print("\nEnter message:")
mes = input()
x = encrypt(mes,len(clientkey[2]),clientkey[0],clientkey[1],clientkey[2])
encmes=pickle.dumps(x)
c.send(encmes)
```

OUTPUT

Client.py



Server.py


C:\Windows\py.exe
Decrypted message: Good evening sir by group 7
Enter message:
I hope you liked our project
Decrypted message: We successfully completed our cryptography project
Enter message:
It was fun
Decrypted message: We learned a lot
Enter message:
22°C Cloudy
ENG
20:28

PROS

- **Multiple rounds of encryption make the algorithm hard to break via brute force attack.**
- **We randomize the initial prime number picked, the number of integers in the private key (number of rounds) whichmakes it hard for the hacker to crack the system.**
- **A new key is generated for each of the participants of theconversation after each round of communication, whichprevents hackers from gaining access to future messageseven if they intercept the keys of one communication.**
- **Since we have modified an existing algorithm in a novelway, hackers who intercept the public key will be confuse das to what algorithm we have used to encrypt our messagessince the structure of our key system is different.**
- **By padding the message with a random number of randomcharacters during encryption we ensure that the attacker cant guess the length of the original message.**
- **By padding the message, we also make it useless for the attacker to try and change the message in anymeaningful way as he doesn't know which part of themessage is a pad and which is the message.**
- **Since we map the characters of the encrypted message back toUnicode, the encrypted message is not displayed in many Englishdevices as the characters aren't supported by the device. This may make it seem like the messages are getting corrupted during transmit but they actually aren't.**

- **Since this is an asymmetric public-key encryption system the attacker cant decode the message without the private key of the recipient of the message.**

Conclusion and Future Work

In this , we presented the modified ElGamal cryptosystemand three features/modifications to it. Since the structure of the new system is similar to the existing Elgamal architecture, it could be used to replace the system with our better one. As for future work, we plan to reduce the computing time because it needs more time to generate random numbers.

References

1. Wu, Jiahui, Xiaofeng Liao, and Bo Yang. "Color image encryption based on chaotic systems and elliptic curve ElGamal scheme." SignalProcessing 141 (2017): 109-124.

2. Minfeng, Fu, and Chen Wei. "Elliptic curve cryptosystem ElGamalencryption and transmission scheme." 2010 International Conference on Computer Application and System Modeling (ICCASM 2010). Vol.6. IEEE, 2010.

3. Siahaan, Andysah Putera Utama. "Comparative analysis ofrsa and elgamal cryptographic public-key algorithms." (2018).

4. Sodiya, Adesina Simon, S. A. Onashoga, and D. I. Adelani."A secure e-voting architecture." 2011 Eighth InternationalConference onInformation Technology: New Generations. IEEE,2011.

5. Ara, Anees, et al. "A secure privacy-preserving data aggregation scheme based on bilinear ElGamal cryptosystem for remote health monitoring systems." IEEE Access 5 (2017):12601-12617.

6. Castagnos, Guilhem, and Benoît Chevallier-Mames. "Towards a DL-based additively homomorphic encryption scheme." International Conference on Information Security. Springer, Berlin, Heidelberg,2007.

7. Hua, Zhuping, and Fuyang Xia. "A new cryptosystem anddigital signature scheme based on ElGamal cryptosystem." 2011 International Conference on Computer Science and Service System (CSSS). IEEE, 2011.

8. Huang, Kaibin, and Raylin Tso. "A commutative encryptionscheme based on ElGamal encryption." 2012 InternationalConference on Information Security and Intelligent Control.IEEE, 2012.

9. Zhang, Jun, Hui Ying Zhang, and Wei Dong Ji. "Notice ofRetraction: ElGamal Digital Signature Scheme with a PrivateKey Pairs." 2010 2nd International Conference on InformationEngineering and Computer Science. IEEE, 2010.

10. Li, Xiaofei, Xuanjing Shen, and Haipeng Chen. "Elgamaldigitalsignature algorithm of adding a random number." Journal of Networks 6.5 (2011): 774.

11. Zhumaniezov, Alisher R. "Research and optimization ofElGamal Encryption System." Contemporary Dilemmas:Education, Politics andValues 6 (2018).

12. El Makkaoui, Khalid, Abderrahim Beni-Hssane, andAbdellah Ezzati. "Cloud-ElGamal: an efficient homomorphicencryption scheme." 2016 International Conference on Wireless Networks and MobileCommunications (WINCOM).IEEE, 2016.