VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

REVIEW - 1

ITE4001 - Network and Information Security

SLOT A1 + TA1

Topic **Optimized Elgamal Encryption**

By

NALLA SANJAY REDDY	**18BIT0100**
MAHITH KUMAR REDDY	**18BIT0076**
SAI RISHITH KUMAR	**18BIT0193**

Introduction

We have created a new encryption technique by modifying the existing Elgamal technique.

ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie–Hellman key exchange. Its security depends upon the difficulty of a certain problem related to computing discrete logarithms.

We modify the standard Elgamal encryption by adding the following features to 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

Proposed encryption system: Modified Elgamal encryption

The following is the process followed by our method for a client to send a message to the server

1. Key generation (Server):

- Choose a random prime number (p) from 11 to 255 and choose a random primitive root(g) of the prime number
- Choose a random integer (n) between 4 and 9 as the number of keys (rounds) in our private key
- Generate n random integers (b) which will act as our private key, apply g**b%p to each of the integers to generate B (list of integers).
- Send p, g, B to client which acts as our public key.

2. Encrypt message (Client):

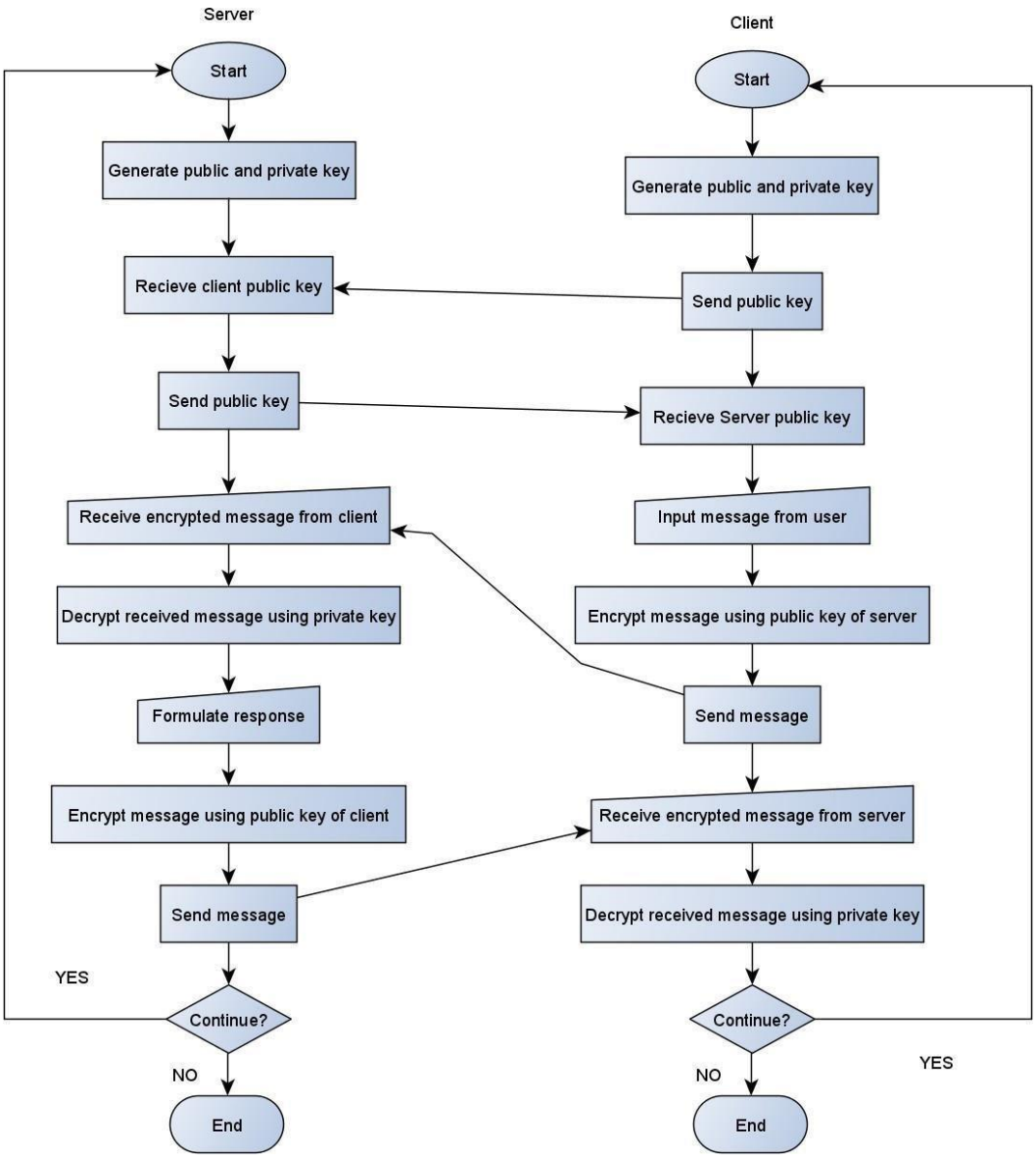
- Receive server's public key (p, g, B)
- Choose length(B) random numbers (a)
- Compute B^a%p using all numbers in B and a and multiply them and store in c and then set c as c%p
- Compute a list A = g^a%p using all integers in a
- Pad the message with c random characters in the beginning
- Encrypt message by multiplying c with the Unicode value of the characters in the message and then convert them back to characters resulting in encrypted message X Send A,X to Server

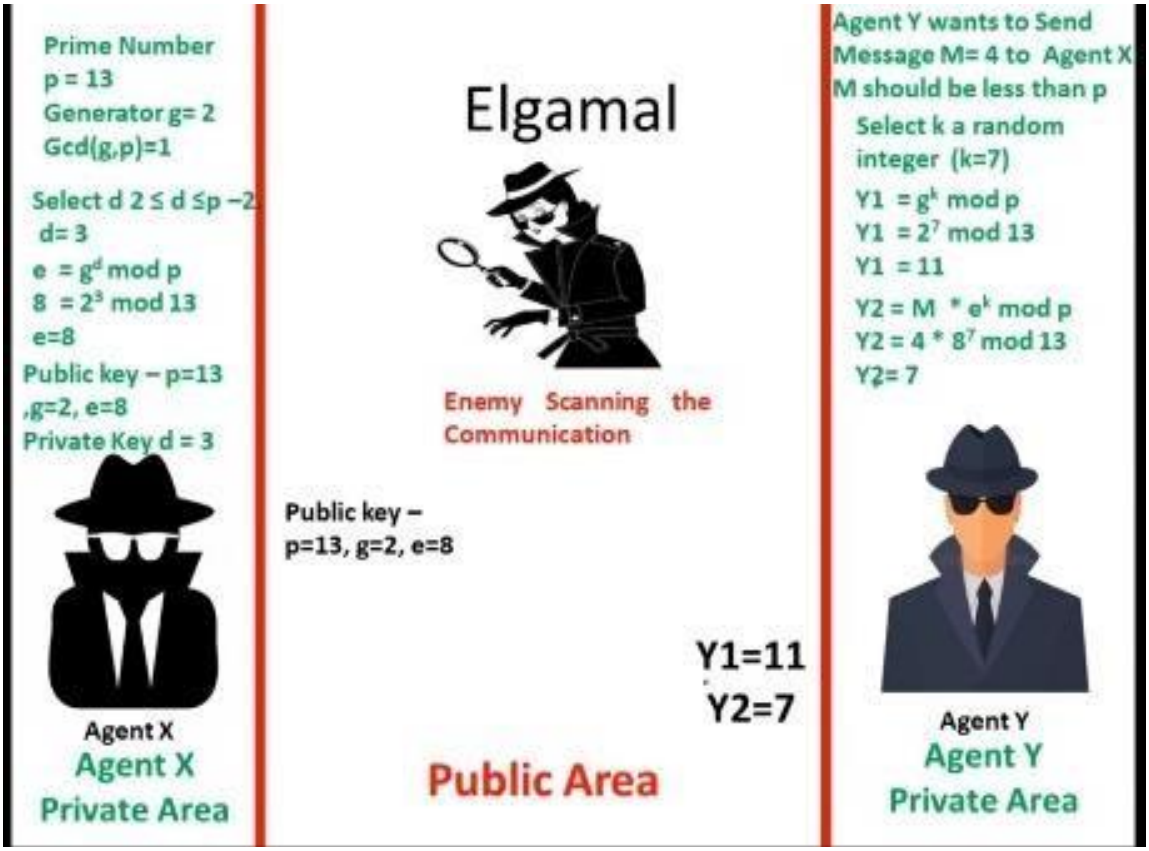
3. Decrypt message (Server):

- Receive A,X from client
- Compute c by applying A**b%p on all the integers in A and b then multiply them together and mod them with p

- Begin processing message from position c as the characters before it are just padding.
- Divide the Unicode value of each character in the message by c and then convert them back to a character.
- This is our decrypted message.

The following is a flow-chart depicting the process of sending and receiving messages using our algorithm.





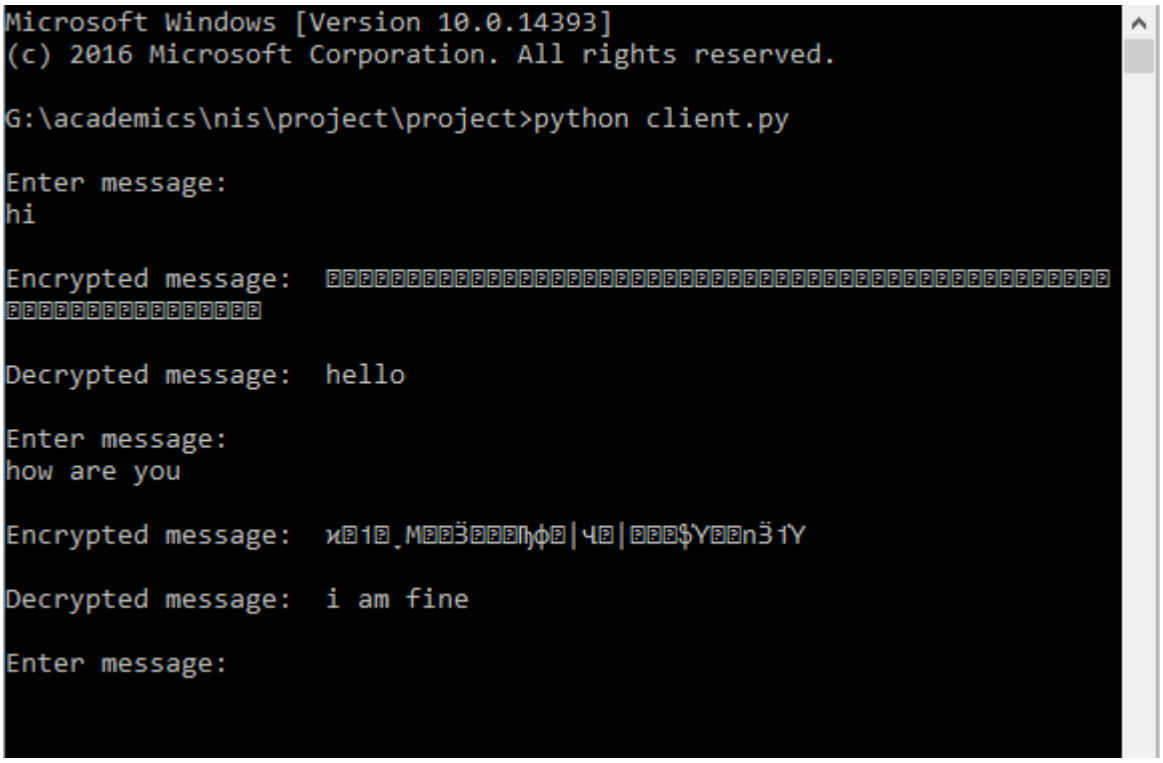
Decryption At sender's side

$$\text{Plain text} = Y2 * (Y1^{d})^{-1} \bmod p$$
$$\text{Plaint text} = 7 * (11^{3})^{-1} \bmod 13$$
$$\text{Plaint text} = 7 * 8 \bmod 13$$
$$\text{Plaint text} = 56 \bmod 13$$
$$\text{Plain text} = 4$$

FULL CODE AND REPORT : https://drive.google.com/file/d/1jn-GUNgrRVCBYAZF6kJmX2w04tkzz-yx/view?usp=sharing

screenshots:

client side screenshot:



Server side screenshot: