CSE1011

CRYPTOGRAPHY FUNDAMENTALS

EL GAMAL CRYPTOSYSTEM

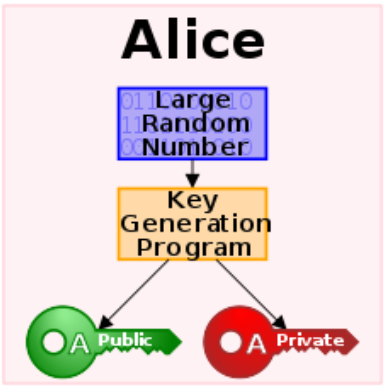
NAME	REGISTRATION NUMBER	SLOT
VOLETI RAVI	15BCE0082	F1
SAMUDRA PRATIM BORKAKOTI	15BCE0093	F2
UJJWAL VERMA	15BCE0571	F2
SAMYAK JAIN	16BCE0712	F2

I. INTRODUCTION

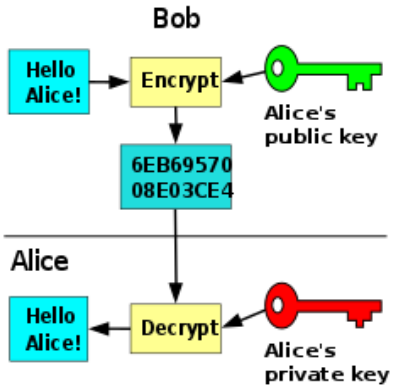
1.1 PUBLIC KEY CRYPTOGRAPHY

Public key cryptography, or asymmetrical cryptography, is any cryptographic system that uses pairs of keys: public keys which may be disseminated widely, and private keys which are known only to the owner. This accomplishes two functions: authentication, which is when the public key is used to verify that a holder of the paired private key sent the message, and encryption, whereby only the holder of the paired private key can decrypt the message encrypted with the public key.

In a public key encryption system, any person can encrypt a message using the public key of the receiver, but such a message can be decrypted only with the receiver's private key. For this to work it must be computationally easy for a user to generate a public and private key-pair to be used for encryption and decryption. The strength of a public key cryptography system relies on the degree of difficulty (computational impracticality) for a properly generated private key to be determined from its corresponding public key. Security then depends only on keeping the private key private, and the public key may be published without compromising security. [1]



[1] generating a public key.



[1] using a public key for encryption and private key for decryption.

1.2 IMPORTANT PROPERTIES

The most important properties of public key encryption that helps in security are:

a) Different keys are used for encryption and decryption. This is a property which set this scheme different than symmetric encryption scheme.

b) Each receiver possesses a unique decryption key, generally referred to as his private key.

c) Receiver needs to publish an encryption key, referred to as his public key.

d) Some assurance of the authenticity of a public key is needed in this scheme to avoid spoofing by adversary as the receiver. Generally, this type of cryptosystem involves trusted third party which certifies that a particular public key belongs to a specific person or entity only.

e) Encryption algorithm is complex enough to prohibit attacker from deducing the plaintext from the cipher text and the encryption (public) key.

f) Though private and public keys are related mathematically, it is not feasible to calculate the private key from the public key. In fact, intelligent part of any public-key cryptosystem is in designing a relationship between two keys. [2]

1.3 EL GAMAL CRYPTOSYSTEM

In cryptography, the ElGamal encryption system is an asymmetric key encryption algorithm for public-key cryptography which is based on the Diffie–Hellman key exchange. The system provides an additional layer of security by asymmetrically encrypting keys previously used for symmetric message encryption. It was described by Taher Elgamal in 1985. ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems. The Digital Signature Algorithm (DSA) is a variant of the ElGamal signature scheme, which should not be confused with ElGamal encryption. [3]

II. ALGORITHM [2]

This encryption has three distinct important components:

i. Key generation
ii. Encryption
iii. Decryption

This document will explain these segments in detailed below with proper explanation under each component.

i. KEY GENERATION

Each user of ElGamal cryptosystem generates the key pair through as follows:

a. **Choosing a large prime *p***. Generally, a prime number of **1024** to **2048** bits length is chosen.

b. **Choosing a generator element *g***.
    a. This number must be between **1** and **p − 1**, but cannot be any number.

b. It is a generator of the multiplicative group of integers **modulo *p***. This means for every integer ***m*** co-prime to a number ***p***, there is an integer ***k*** such that $g^k = a \bmod n$.

c. **Choosing the private key**. The private key ***x*** is any number bigger than **1** and smaller than ***p*−1**.

d. **Computing part of the public key**. The value ***y*** is computed from the parameters ***p, g*** and the private key ***x*** as follows:

$$y = g^x \bmod p$$

e. Obtaining Public key. The ElGamal public key consists of the three parameters **(p, g, y)**.

For example, suppose that ***p*** = 17 and that ***g*** = 6 (It can be confirmed that 6 is a generator of group Z_{17}). The private key ***x*** can be any number bigger than 1 and smaller than 71, so we choose ***x*** = 5. The value ***y*** is then computed as follows:

$$y = 6^5 \bmod 17 = 7$$

Thus the private key is 62 and the public key is **(17, 6, 7).**

ii. ENCRYPTION

The encryption and decryption is similar to RSA algorithm but slightly more complex.

Suppose sender wishes to send a plaintext to someone whose ElGamal public key is ***(p, g, y)***

a. Sender represents the plaintext as a series of numbers modulo p.
b. To encrypt the first plaintext P, which is represented as a number modulo p. The encryption process to obtain the cipher text C is as follows −
    a. Randomly generate a number k.
    b. Compute two values C1 and C2 such that:

$$C1 = g^k \bmod p$$

$$C2 = (P^* y^k) \bmod p$$

c. Send the cipher text ***C***, consisting of the two separate values ***(C1, C2)***, sent together.
d. Contd. Example

Let the plain text ***P*** = 13 be encrypted

We randomly generate a number, ***k*** = 10 (say) and compute the value of ***C1*** and ***C2*** such that

$$C1 = 6^{10} \bmod 17 = 15$$

$$C2 = (13^* 7^{10}) \bmod 17 = 9$$

Therefore cipher ***C*** = ***(C1, C2)*** => (15, 9)

iii. DECRYPTION

To decrypt the cipher text ***(C1, C2)*** using private key x, the following two steps are taken

a. Compute the modular inverse of $(C1)^x$ modulo ***p***, which is $(C1)^{-x}$, generally referred to as decryption factor.
b. Obtain the plaintext by using the following formula:

$$C2 \times (C1)^{-x} \bmod p = Plaintext$$

Contd. Example

To decrypt the example (15,9) using private key ***x*** = 5, the decryption factor is:

15^{-5} **mod** *17* **=** *9*

Extract plaintext ***P*** **=** *(9 × 9)* **mod** *17* = 13.

III. RSA AND ELGAMAL SCHEMES – A COMPARISON

RSA	ELGAMAL
It is more efficient for encryption.	It is more efficient for decryption.
It is less efficient for decryption.	It is less efficient for encryption.
For a particular security level, lengthy keys are required in RSA.	For the same level of security, very short keys are required.
It is widely accepted and used.	It is new and not very popular in market.

Comparing RSA and Elgamal encryption [2]

IV. ELGAMAL SUMMARY

As has been shown, the ElGamal cryptosystem is as secure as it is hard to solve the Discrete Logarithm problem, given no weak random exponents or primes are chosen. It further prevents a chosen plaintext attack by using a randomized encryption exponent k. As this k is chosen uniformly before encryption, the same plaintext can result in p−1 different cipher texts, one of which is chosen uniformly by choosing a k. One shortcoming of ElGamal is the Message Expansion. The size of the transferred message increases by the factor 2. This comes from choosing a new random k for each block of the plaintext message m_i . The public key derived from this k has to be sent together with the ci as the pair (ci, g^k), the set of all these pairs giving the cipher text C. Another problem that can arise is

that Alice relies on the authenticity of the public key she retrieves from Bob. This is a lesser problem if the public key is handed over in direct contact, but using the system on a large network like the Internet other means have to be found. The most common installation is a central key server. Users send their public keys there after generation so others can download them and use them for encryption or signature validation. If a malicious attacker manages to supply Alice with his key and make her believe it is the key of Bob, she would encrypt the message to the attacker and not to the intended receiver Bob. If the message is not signed, the attacker can then encrypt the message again, this time with Bob's key and send it to Bob. This is called the Man-In-The-Middle attack. At least the man in the middle would be detected if Alice would both sign and encrypt the message, since Bob would notice the changed contents when verifying the signature. To overcome the problem of forged signatures, webs of thrust can be built up where users sign their keys against each other thereby assuring the authenticity of the key. Another possibility is a central certification authority that signs or even gives out the keys only after validating the owner of the key. In this section, we have presented the ElGamal cryptosystem as presented in 1984 and followed the algorithm through the intended steps. The next section will explain the problems that can arise during the implementation of cryptography software and show an example where an implementation error has long lived undetected. [4]

V. OBSERVATION AND RESULTS

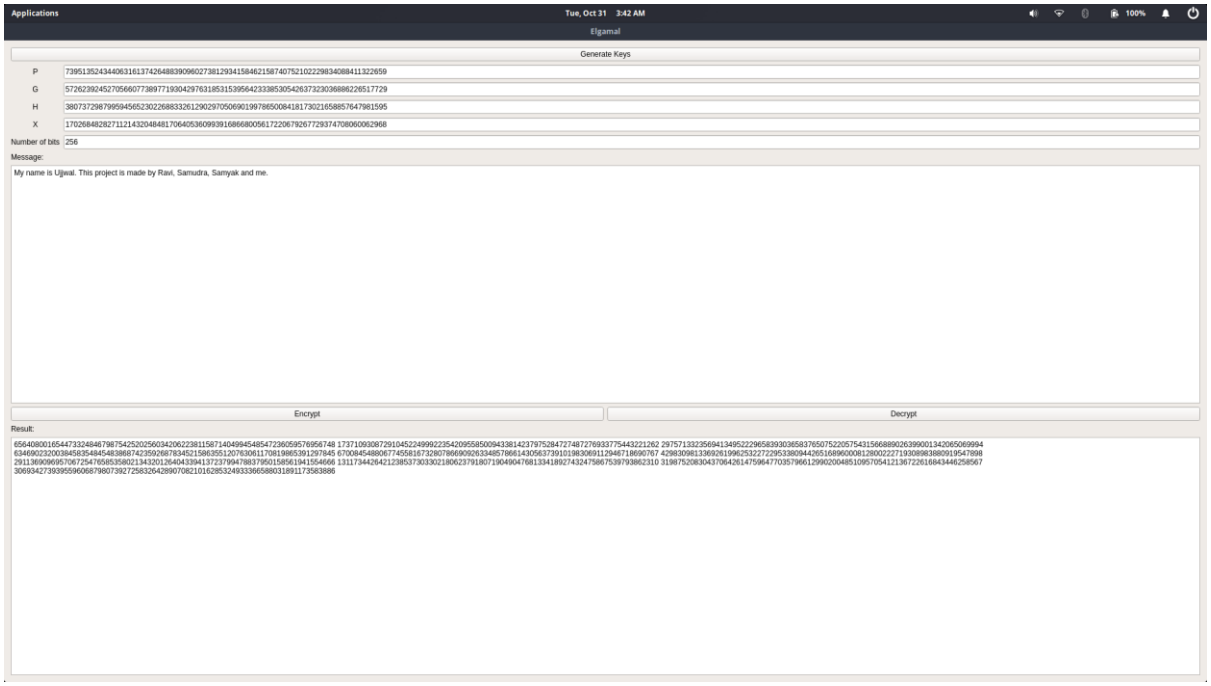


Fig 1. ENCRYPTION

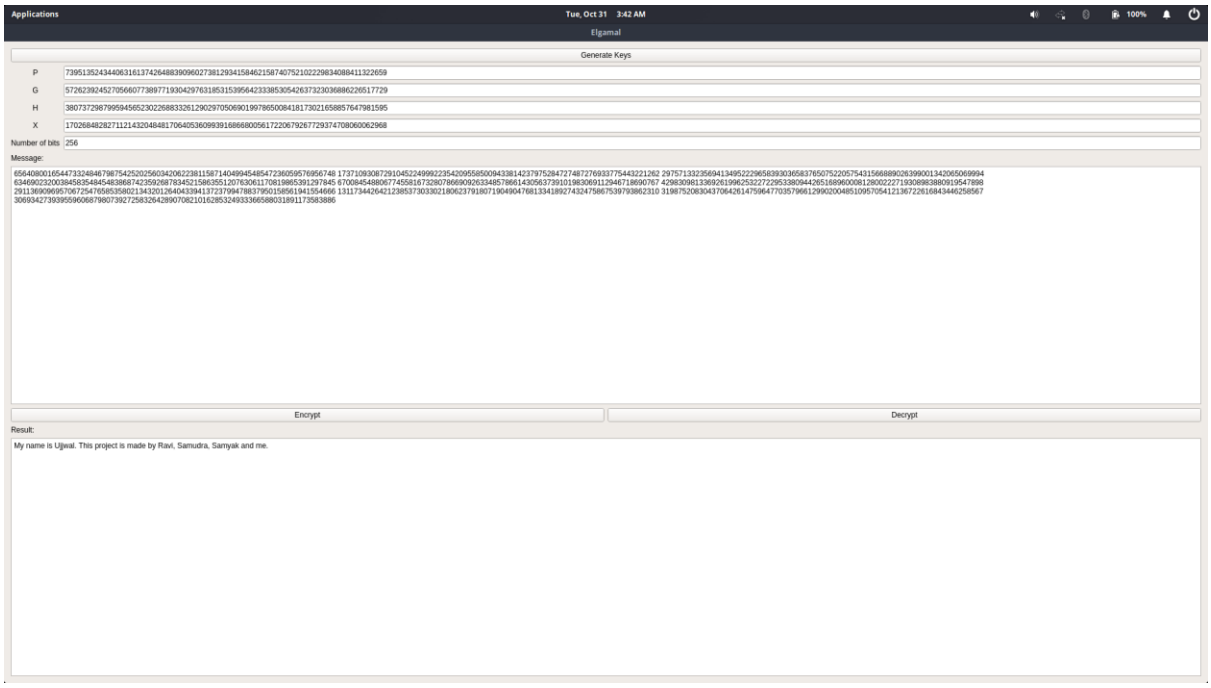


Fig 2. DECRYPTION

REFERENCES:

[1] https://en.wikipedia.org/wiki/Public-key_cryptography

[2] https://www.tutorialspoint.com/cryptography/public_key_encryption.htm

[3] https://en.wikipedia.org/wiki/ElGamal_encryption

[4] http://wwwmayr.in.tum.de/konferenzen/Jass05/courses/1/papers/meier_paper.pdf