

Laboratory 5

Problem : Public-key cryptography (ElGamal)

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. It was described by Taher Elgamal in 1984.[1] ElGamal encryption is used in the free GNU Privacy Guard software, recent versions of PGP, and other cryptosystems. The Digital Signature Algorithm is a variant of the ElGamal signature scheme, which should not be confused with ElGamal encryption.

ElGamal encryption can be defined over any cyclic group G . Its security depends upon the difficulty of a certain problem in G related to computing discrete logarithms.

ElGamal encryption consists of three components: the key generator, the encryption algorithm, and the decryption algorithm.

Key generation

The key generator works as follows:

- Alice generates an efficient description of a multiplicative cyclic group G of order q with generator g . See below for a discussion on the required properties of this group.
- Alice chooses a random x from $\{0, \dots, q-1\}$.
- Alice computes $h = g^x$.
- Alice publishes h , along with the description of G, q, g , as her **public key**. Alice retains x as her **private key** which must be kept secret.

Encryption

The encryption algorithm works as follows: to encrypt a message m to Alice under her public key (G, q, g, h) ,

- Bob chooses a random y from $\{0, \dots, q-1\}$, then calculates $c_1 = g^y$.
- Bob calculates the shared secret $s = h^y$. Since a new y is generated for every message^[why?], y is also called an ephemeral key.

The steps above can be computed ahead of time.

- Bob converts his secret message m into an element m' of G .
- Bob calculates $c_2 = m' \cdot s$.
- Bob sends the ciphertext $(c_1, c_2) = (g^y, m' \cdot h^y) = (g^y, m' \cdot (g^x)^y)$ to Alice.

Decryption

The decryption algorithm works as follows: to decrypt a ciphertext (c_1, c_2) with her private key x ,

- Alice calculates the shared secret $s = c_1^x$
- and then computes $m' = c_2 \cdot s^{-1}$ which she then converts back into the plaintext message m .

The decryption algorithm produces the intended message, since

$$c_2 \cdot s^{-1} = m' \cdot h^y \cdot (g^{xy})^{-1} = m' \cdot g^{xy} \cdot g^{-xy} = m'.$$

The ElGamal cryptosystem is usually used in a hybrid cryptosystem. I.e., the message itself is encrypted using a symmetric cryptosystem and ElGamal is then used to encrypt the key used for the symmetric cryptosystem. This allows encryption of messages that are longer than the size of the group G .

In our case, we have applied the algorithm as follows: each letter from the input message is converted to a numeric form, representing the letter's position in the alphabet, and then it is encrypted. When decrypting, the algorithm is applied in the same manner.

Later on, we present a running example of our application.

Firstly, a key is generated (we only give the private key manually).

The screenshot shows a Java Swing window titled "Form 1" with a standard Windows-style title bar (minimize, maximize, close buttons). The window contains a tabbed interface with three tabs: "Keygen", "Encryption", and "Decryption". The "Keygen" tab is currently selected. Inside the "Keygen" tab, there is a text input field containing the value "740653". Below this field is a button labeled "Generate Random Prime Number". Further down, there is a label "Private Key :" followed by a text input field containing the value "1742". Below that is a label "Public Key :" followed by a larger text area containing the value "740653 2 273945". At the bottom center of the window is a button labeled "Generate !".

Then, the message is encrypted.

Form1

Keygen Encryption **Decryption**

Public Key : 740653 2 273945

K = 13

Message : roses are violets and red are blue

Cipher Text : 8192 95930 326826 265849 108942
265849 0 169919 95930 108942 0 34953
47965 326826 557722 108942 435768
265849 0 169919 156907 679676 0
95930 108942 679676 0 169919 95930
108942 0 339838 557722 605687
108942

Generate !

Using the same key, the message is decrypted.

Form1

Keygen Encryption **Decryption**

Private Key : 1742

Prime Number : 740653

Cipher Text : 8192 95930 326826 265849 108942
265849 0 169919 95930 108942 0
34953 47965 326826 557722 108942
435768 265849 0 169919 156907
679676 0 95930 108942 679676 0
169919 95930 108942 0 339838
557722 605687 108942

Message : roses are violets and red are blue

Generate !