ElGamal Cryptosystem

Like RSA, ElGamal is a *public key cryptosystem:* The encryption key is published, and the decryption key is kept private. This allows an entity (human or computer) to receive encrypted messages from diverse senders with reasonable confidence that the senders' messages cannot be decrypted by anyone other than the intended recipient.

Since encryption and decryption are inverse proceedures, there must be a mathematical relationship between the encryption and decryption keys. Security in public key cryptosystems relies on this relationship being one that cannot easily be exploited to deduce the (private) decryption key from knowledge of the (public) encryption key: The underlying mathematical problem that would produce the decryption key from the encryption key must be computationally infeasible to solve.

In ElGamal, the underlying mathematical relationship between the encryption and decryption keys relies upon the so-called *discrete log problem,* which will be described a bit later.

To create a key for ElGamal one uses a so-called *primitive root.* The theory of primitive roots (see any text on elementary number theory) guarantees that if p is a prime number, then there is some integer $\alpha \in \{1, 2, \ldots, p-1\}$ such that all of the remainders $\alpha^t \pmod{p}$, for $t = 1, \ldots, p-1$, are distinct. Such an integer α is called a *primitive root modulo* p. For example, 2 is a primitive root modulo 5, since $2^1 \pmod{5}$, $2^2 \pmod{5}$, $2^3 \pmod{5}$, and $2^4 \pmod{5}$ are distinct, but 4 is not a primitive root modulo 5, since $4^2 \equiv 4^4 \equiv 1 \pmod{5}$.

Observe that if α is a primitive root modulo p, then the integer powers of α, when reduced modulo p, comprise all of the possible remainders modulo p, except 0. Hence every integer that is not divisible by p is congruent modulo p to some power of α.

An ElGamal encryption key is constructed as follows. First, a very large prime number p is chosen. Then a primitive root modulo p, say α, is chosen. Finally, an integer a is chosen and $\beta = \alpha^a \pmod{p}$ is computed. The encryption key is the ordered triple (p, α, β). The encryption key (p, α, β) is made public, HOWEVER, the integer a that was used to create β is kept secret. It is not needed to encrypt messages, but will be needed to decrypt them. (The *decryption key* is a.)

To send an encrypted message, the plaintext first must be converted to a numerical representation. As with RSA, there are several concerns when doing this. For our purposes, we shall assume that this conversion already has been done, resulting in a (potentially quite large) positive integer m. We shall also assume that $m < p$. Normally, this is the case; if it is not, there would need to be a mutually agreed-upon method to modify the encryption process, (perhaps partitioning m in some way and then encrypting the message in "packets").

Suppose Bob has ElGamal public key (p, α, β), and Alice wants to use ElGamal to encrypt a message m and send it to Bob. Once she has looked up Bob's public key, here is how Alice would proceed.

i. She chooses a secret integer k and computes $r \equiv \alpha^k \pmod{p}$.
ii. Using the same secret integer k, she also computes $t \equiv \beta^k m \pmod{p}$.
iii. The ordered pair $c = (r, t)$ is Alice's encrypted message. The integer k is known only to Alice. It is not needed for decryption, and she does NOT send it to Bob.

As with RSA, one does not substitute characters for c. Consequently, we refer to c itself as the "ciphertext."

Once Alice's encrypted message has been transmitted to Bob,

- Alice knows $p, \alpha, \beta, m, k, r, t$ but not a.
- Bob knows $p, \alpha, \beta, a, r, t$ but not k. Using a, Bob can compute $m \equiv tr^{-a} \pmod{p}$.
- An eavesdropper knows p, α, β, r, t but not m, k, a. Knowledge of either a or k would be enough to compute m, since $m \equiv tr^{-a} \equiv t\beta^{-k} \pmod{p}$, so it is important that a and k are both kept secret.

For practice, we'll encrypt $m = 138$ using ElGamal with encryption key $(257, 3, 112)$. Let us choose $k = 72$ as our secret integer. We find $r \equiv 3^{72} \equiv 137 \pmod{257}$ and $t \equiv 112^{72}138 \equiv 229 \pmod{257}$. Hence our ciphertext is the pair $(137, 229)$.

Suppose someone wants to attack the ElGamal encryption key $(257, 3, 112)$. This means they want to find the secret integer a such that $3^a \equiv 112 \pmod{257}$. If this congruence were an equation, its solution would be obtained using a logarithm. (Of course, if one computes $\log_3 112$, one does not obtain the integer a; the real number $\log_3 112$ is roughly 4.295, so not an integer at all!) The solution a to the congruence must be an integer, and hence cannot be obtained so quickly. The problem of solving such a congruence is called a discrete log problem.

There is no known method for solving a discrete log problem with a large prime modulus that is sufficiently efficient to be practical for application to cryptanalysis. Hence the ElGamal cryptosystem is thought to be secure for sufficiently large prime modulus.

The modulus in our sample discrete log problem is only 257, so by trial and error one may quickly find that $3^a \equiv 112 \pmod{257}$ has solution $a = 21$. Knowing $a = 21$ allows the recipient of our ciphertext message, (or anyone who may have intercepted it), to compute $(229)137^{-21} \pmod{257}$, which yields $m = 138$.

Our would-be attacker would have no better luck attacking a particular ciphertext. For that, they would want to find the secret integer k such that $r \equiv \alpha^k \pmod{p}$, another discrete log problem!

The ElGamal cryptosystem was invented in 1985, by Taher Elgamal. Contemporary elliptic curve cryptography (ECC) is an analogue of ElGamal that uses the group of points on an elliptic curve in place of the integers.