

The **ElGamal public key cryptosystem** is defined as follows: Fix a large prime p which is public. Also public is a primitive root g modulo p in $1 < g < p$. Each user A who wishes to participate in this public-key cryptosystem chooses a secret a_A in $0 < a_A < p - 1$ and publishes $b_A = g^{a_A} \bmod p$. When a user B wants to send a secret message M in $0 < M < p$ to A , she chooses a random k in $0 < k < p - 1$ and sends to A the pair

$$C = (g^k \bmod p, (Mb_A^k) \bmod p).$$

The plaintext M is enciphered by multiplying it by b_A^k in the second component of C . Note that $b_A^k \equiv (g^{a_A})^k \equiv g^{a_A k} \pmod{p}$. The first component of C provides a hint for deciphering M from the second component of C , but one which is useful only to A . Only A knows the secret key a_A , so only A can compute $(g^k)^{a_A} \equiv g^{a_A k} \pmod{p}$. If the multiplicative inverse of this number is multiplied times the second component, one recovers M :

$$(g^{a_A k})^{-1} (M b_A^k) \equiv (g^{a_A k})^{-1} (M g^{a_A k}) \equiv M \pmod{p}.$$

An eavesdropper who could solve the discrete logarithm problem modulo p could compute M from C and public data without knowing a_A as follows. The first component of C is $h = g^k \pmod{p}$. This number and $T = (Mb_A^k) \pmod{p}$ are observed by the eavesdropper. The eavesdropper knows p and g because these numbers are public. He can also obtain A 's public key b_A from A 's directory, just as B did. He would solve the discrete logarithm problem $g^k \equiv h \pmod{p}$ for k and then compute

$$T (b_A^k)^{-1} \equiv (Mb_A^k) (b_A^k)^{-1} \equiv M \pmod{p}.$$