

CS3441/9441 Tutorial 3

ElGamal Public-Key Algorithm

ElGamal keys are generated as follows: First choose a prime p , and two random numbers, g and x , such that both g and x are less than p . Both g and p can be made public. Then calculate

$$y = g^x \pmod{p}$$

The public key is $\langle y, g, p \rangle$. The private key is x .

To encrypt a message M , first choose a random k , such that k is relatively prime to $p - 1$. Then compute

$$a = g^k \pmod{p}$$

$$b = y^k M \pmod{p}$$

The ciphertext is $\langle a, b \rangle$. To decrypt, compute

$$M = b/(a^x) \pmod{p}$$

1. Show that ElGamal encryption works, i.e. show that decryption is the inverse of encryption.
2. Argue why this algorithm is secure. (Note that we are not asking you to prove this!)
3. This is similar to a key exchange algorithm you have learned. Which algorithm is this, and why are they similar?
4. ElGamal signatures are generated as follows (the message is M):
 - (a) Generate a random number k relatively prime to $p - 1$
 - (b) Compute $a = g^k \pmod{p}$.
 - (c) Solve $M = (xa + kb) \pmod{(p - 1)}$ for b . (Which algorithm would you use?)
 - (d) The signature is $\langle a, b \rangle$.

How can this signature be verified?

5. It is necessary to choose a new value of k for each signature. Why?
6. What properties should digital signature algorithms satisfy?