CSEP 590A: Practical Aspects of Modern Cryptography Winter 2011

Assignment #8

Due: March 10, 2011

As described in class, threshold encryption allows a group of n participants to jointly generate a public key encryption function E such that any k of the n participants can jointly decrypt any message $E(m)$ while fewer than k cannot determine anything about m. While some threshold encryption schemes require a trusted dealer or cannot be used more than once, we will see here that threshold ElGamal allows the n participants to jointly derive a public key without outside help and such that any k of these n participants can decrypt an encrypted message $E(m)$ by generating shares of the decryption that can be combined to derive m without revealing anything about their keys.

To begin, several parameters are agreed upon. These may be public values that are used repeatedly by many groups wanting to establish a threshold cryptosystem. Large primes p and q are selected such that $p-1$ is a multiple of q (it is common to use $p = 2q + 1$). A value h with $0 < h < p$ is then selected such that $g = h^{(p-1)/q} \bmod p \neq 1$. This g has the property that $g^q \bmod p = 1$ since Fermat's Little Theorem tells us that $g^q \bmod p = h^{q(p-1)/q} \bmod p = h^{p-1} \bmod p = 1$. Arithmetic within exponents of g now works modulo q while arithmetic in the base is performed modulo p. (Although this may seem complex, the moduli do not cause difficulties here as long as they are used as described here.)

Once the general parameters are fixed, each of the n participants forms a degree k (or smaller) polynomial with coefficients less than q. Specifically, if the n keyholders are designated as $T_1, T_2, \dots, T_n$, then each keyholder T_i selects $a_{i,0}$, $a_{i,1}$, …, $a_{i,k-2}$, $a_{i,k-1}$ with each $a_{i,j}$ in the range $0 \leq a_{i,j} < q$ and forms the polynomial

$$P_i(x) = a_{i,k-1}x^{k-1} + a_{i,k-2}x^{k-2} + \cdots + a_{i,1}x + a_{i,0}.$$

[Note that if a value $a_{i,k-1}$ happens to be chosen to be equal to 0, the degree of P_i will be less than k.]

Each keyholder T_i then forms and publishes commitment values $g_{i,j} = g^{a_{i,j}} \bmod p$ for each $a_{i,j}$ ($0 \leq j < k$) and explicitly gives the value $P_i(l)$ to each other keyholder T_l who can verify this value by computing and checking that $g^{P_i(l)} \bmod p = \prod_{j=0}^{k-1} {g_{i,j}}^{l^j} \bmod p$. The joint private key is now $S = \sum_{i=1}^{n} a_{i,0} \bmod q$ and the corresponding joint public key $Z = \prod_{i=1}^{n} g_{i,0} \bmod p$. Each of the n participants T_i has data which can be used to compute a share of the private key. A message m can be encrypted for the group by selecting a random value c in the range $0 < r < p$ and forming the encryption $(X, Y) = (mZ^r \bmod p, g^r \bmod p)$.

1. Describe how each of the n group members uses the data it has to form a share of the group private key $= \sum_{i=1}^{n} a_{i,0} \mod q$.

2. Given an encrypted message (X, Y) describe how each member T_i in any set of k group members can (with knowledge of all group members' identities) compute a value V_i such that all k of the V_i values can be combined with (X, Y) to compute its decryption. [Hint: When using Lagrange interpolation to reconstruct a secret, only the constant term of each of the derived polynomials needs to be used.]

3. Suppose that you are given a set of ElGamal encryptions that you do not have the means to decrypt. Describe how you can create a new set of encryptions of the same plaintext messages and, without revealing the correspondences between individual ciphertexts, provide a *strong* interactively prove (not cut and choose, but one that has exponentially small probability of error) that the new set consists of encryptions of exactly the same plaintext messages as the original set. [Hint: You may need to construct *many* sets of encryptions of the same plaintext messages.]