Advanced Topics in Cryptography

Homework 1

Due by April 22, 2006 (before class).

Prove the correctness of all your answers.

1. Consider the following problem: For given values of *n*, *g*, and *w*, compute $[w]_g$ in Z_{n^2}.
   a. Show that if, for given *n* and *g* values, it is possible to solve this problem for a randomly chosen *w* with probability better than ½, then it is easy to solve this problem for any value of *w*.
   (This argument shows random self reducibility. Namely, that there is no *w* value for which this problem is harder than for a randomly chosen *w* value. The same argument can be shown to be true if ½ is replaced with 1/p(n), for any polynomial p()).
   b. Show that if there is an efficient algorithm for solving this algorithm for n and g_1, then for any g_2 there is also an efficient algorithm for solving this algorithm for n and g_2.
   (This argument shows that there is no *g* value for which this problem is harder than for other *g* values.)

2. Consider the following problem. A client has a list of n items, $X = x_{1,}\ldots,x_n$, a server has a list of n items, $Y = y_1,\ldots,y_n$. The two parties have to run a protocol, at the end of which the client learns the intersection of the two lists, and nothing else, and the server learns nothing.
   At the first step of the protocol the client defines a polynomial P(y) of degree k whose roots are her inputs $x_1,\ldots,x_k$.
   Namely, $P(y) = (x1-y)(x2-y)\ldots(xk-y) = a_0 + a_1 y + a_2 y^2 +\ldots+ a_k y^k$.
   The client uses a Homomorphic encryption system whose decryption key is known only to her, and sends to the server encryptions of the coefficients of the polynomial, $E(a_0)$, $E(a_1)$,…, $E(a_n)$.
   a. Show how for any y_i value, the server can compute $E(P(y_i))$.
   b. In the protocol the server chooses, for any y_i, a random value r_i, and computes $E(r_iP(y_i) + y_i)$. Show how the server can compute this value.
   c. The server sends to the client these n values in permuted order. Show how the client can compute $X \cap Y$ from this information. Why is it important for the server to permute the order of the results before sending them?
   d. Argue about the security of the client and server in this protocol.
   e. Suppose that at the end of the protocol the client should only learn the *size* of the intersection (the server should learn nothing). How can the protocol be modified to support this output?