Cryptography W4995
Instructor: Prof. Michael Rabin

Problem Set 4

Due: Tuesday April 10, 2007

1. **(Improving the efficiency of a signature scheme)** Consider the signature scheme in which the public key is a modulus N, the secret key is a pair of primes p, q such that $pq = N$, and the signature on a message M is a value x such that $x^2 = M \bmod N$ (the signer picks one of the square roots of M at random). Note that signature verification involves computing $x^2 \bmod N$ and checking whether this is equal to M; thus, the cost of signature verification is one multiplication over the integers $(x \cdot x)$ and one division over the integers (in order to compute $x \cdot x \bmod N$).

   Assume division takes longer than multiplication. Suggest a way to modify the scheme so that signature verification requires only two multiplications (over the integers). *Hint:* include some extra information with every signature to make verification easier.

2. **(Key agreement in the symmetric private-key model)** To avoid the $\mathcal{O}(N^2)$ blowup in the number of keys required for secure communication in a network of N parties, the Kerberos protocol was suggested. Here, there is a trusted party K with whom every party in the network shares a symmetric encryption key (so that user i shares key e_i with K). When two parties i and j wish to communicate, K helps them to generate a key $e_{i,j}$. All communication between i, j, and K occurs over an insecure channel. Design a secure protocol for doing this and argue why your protocol is secure. Assume a system where all the parties use AES (Advanced Encryption Standard) in which a common private key is needed for all parties (such as i and j) to communicate securely.

   Note that at the beginning of the protocol, user i (respectively, j) is not sure that he is indeed talking to user j (respectively, i) nor that he is indeed talking to K. Similarly, K is not initially sure that he is talking to i or j. The protocol should remain secure even if it is executed many times (not just once). *Hint:* use random strings (*nonces*) which are generated "fresh" each time the protocol is executed.

3. **(Signatures can be pre-computed)** Show that DSS signatures, as explained in class, can also be pre-computed in a manner similar to the pre-computation of signatures for the new signature algorithm presented in class.