Homework II, Foundations of Cryptography 2008

Due on March 7 at 13.15. The general rules on homework solutions available on the course homepage apply. In particular, discussions of ideas in groups of up to at most three people are allowed but implementation should be done individually. Note that there are two types of problems (T(heory) and P(ractice)). Bonus points are counted as usual points but the corresponding problems might be more challenging. How points translate into grades is described in the course memo.

1 (15T) Consider the discrete logarithm problem in Z_p^* with generator g. The complexity of the problem depends heavily on the modulus p but less on the generator g. Suppose a not very sophisticated user only checks that p is prime and we want to look into the dangers of such a behavior. Let us assume that g is indeed a generator in the group Z_p^* (although this might be hard to check).

Describe how a malicious person can choose a large p such that the discrete logarithm is easy. To be more precise, analyze what size p you can construct such that the problem is solvable on an ordinary computer in one hour? Next estimate what happens if p is chosen properly. Here p is chosen to be of the form $1 + 2q$ for a prime q and you should analyze how large values of p are solvable today by using 100 computers during a year. For this second part of the problem you may look for implementation reports the Internet but the construction of an easy p you should do on your own.

2 (15P+3T) Implement Schnorr's signature scheme. Choose the parameters of size so that you expect them to withstand attacks for at least 20 years from attackers willing to spend a total of 10^7 USD. A sound motivation for your parameters gives you the 3T. Choose all parameters needed (public and private for one user) and sign the message "Send more money". In this problem you are allowed to use pre-made routines for arithmetic of large numbers, but not for more advanced operations such as modular exponentiation. You are allowed to use a premade routine for a hash-function you want to use.

3 (20P) Let SHA-1k be a truncated variant of SHA-1 where each of the words used is a k bit integer instead of the designed 32 bit integers. All constants are truncated to the least k significant bits and the basic round function maps $16k$ bits to $5k$ bits. Replace the left shift of 30 by a right shift of 2 keeping all the other operations. Find a collision for this round function. Use as large a value of k as possible. Your score for a correct solution is $\min(4 \cdot (k-8), 20)$. In other words solutions for $k > 13$ do not automatically give extra points but on top of the standard score an additional 40 points are distributed equally among the students solving the largest value of k[1]. The implementation will not be checked against a reference implementation and any implementation that is close enough to give "essentially" SHA-behavior is accepted.

[1]Only solutions handed in on time count with respect to this competition.

4 (12P+8T) Let us consider elliptic curves.

4a (8T) Study the curve defined by

$$y^2 \equiv x^3 + 2x + 6 \bmod 7$$

by hand. Calculate all points and a complete addition table.

4b (12P) Let p_0 be a six digit number giving your date of birth in the order YYMMDD and let p be the smallest prime, $p \geq p_0$ and let q be the smallest prime $q > p$. Find an elliptic on the form

$$y^2 \equiv x^3 + Ax + B \bmod p$$

such that the corresponding elliptic curve group has order q. Find a point on the curve that generates the full group. Motivate your answer why it generates the full group. If you cannot find a curve with order q try to get as close as possible and describe your efforts.

Hint: You might simply try random A and B and do not forget the point at infinity when counting points.

5 (10T+5P) This problem is to study the period of non-linear pseudorandom generators. Let p be a prime and g a generator mod p. Consider the two generators given by x_0 random, and $x_{i+1} = x_i^2 + 4711 \bmod p$ and $x_{i+1} = g^{x_i} \bmod p$ respectively.

5a (5P) Observe by running experiments the periods of these generators by running some tests for reasonable size p (say a few examples for some p of sizes around 10^6, 10^7 and 10^8).

5b (10T) Give guesses how the periods of the generators depend on p (on the average) in the two cases and give a heuristic argument that supports your guesses.

You may freely use any pre-made routines in the experimentation for this problem.

6 (15T) Suppose n is a product of two primes, and x is a number that is relatively prime with n, i.e. $\gcd(x, n) = 1$. Consider the following interactive protocol for proving that x is a non-square modulo n, i.e. that the equation $y^2 \equiv x \bmod n$ is not solvable.

1. The verifier picks a random bit b and a random number r, $1 \leq r \leq n$ that is relatively prime with n. If $b = 0$ then the verifier sends $r^2 \bmod n$ to the prover and if $b = 1$ it sends $xr^2 \bmod n$.
2. The prover responds with a bit b'.
3. The verifier accepts iff $b' = b$.

Prove that this protocol is complete and sound. To prove that it is complete you should prove that if x is a non-square then an all powerful prover can convince a verifier of this fact. Prove that it is efficient in that if the prover knows the factorization of n then it can be implemented efficiently. To prove that it is sound you need to prove that a prover cannot fool an honest verifier too often. To be more precise prove that if x is a square modulo n then, for any strategy of the prover, the probability that a verifier following the protocol accepts is bounded by $1/2$.

Finally discuss whether this protocol has the zero-knowledge property. Argue that if the verifier follows the protocol then in fact it learns nothing new. Discuss what can be said if the verifier deviates from the protocol. Can you see any way to make it zero-knowledge?