COS 433 — Cryptography — Homework 7.

Boaz Barak

Total of 140 points. Due March 31, 2010.

For this exercise, let us say that $\{f_e\}$ is collection of trapdoor permutations if for every $e \in \{0,1\}^n$, f_e is a permutation of $\{0,1\}^n$, there is a polynomial-time algorithm G that on input 1^n outputs a pair (e,d) such that the maps $(x,e) \mapsto f_e(x)$ and $(y,d) \mapsto f_e^{-1}(y)$ can be computed in polynomial time, and for every polynomial-time A there is a negligible function ϵ such that

$$\Pr_{\substack{(e,d)\leftarrow_R G(1^n) \\ x\leftarrow_R\{0,1\}^n}}\left[A(1^n,e,f(x))=x\right] < \epsilon(n)$$

(This is a slight strengthening of the definition of trapdoor functions we saw in class and is in the Boneh-Shoup book, made mostly for simplicity.)

Exercise 1 (30 points)**.** Consider the following public key encryption scheme based on any family of trapdoor permutations $\{f_e\}$.

Key generation Choose $(e,d) \leftarrow_R \mathsf{Gen}(1^n)$ where Gen is the generator for the trapdoor permutation family $\{f_e\}$.

Encryption To encrypt a bit $b \leftarrow_R \{0,1\}$ using the key e: choose $x \leftarrow_R \{0,1\}^n$, choose $r \leftarrow_R \{0,1\}^n$, and output $f_e(x), r, \langle x,r\rangle \oplus b$.

Decryption To decrypt the message (y,r,c) using d: compute $x = f_e^{-1}(y)$ and output $\langle x,r\rangle \oplus c$.

Prove that if $\{f_e\}$ is a trapdoor permutation collection, the above scheme is a CPA secure public key encryption scheme, where the definition of CPA secure public key encryption scheme is the same as the definition of CPA-security for private key schemes, except that the adversary gets the public key e.

Exercise 2 (20 points)**.**

1. We say that a number $y \in Z_n^*$ is a Quadratic Residue (QR) if $y = x^2$ for some $x \in Z_n^*$. (We refer to x as a sqrt of y.) Prove that the set of QRs is a subgroup of Z_n^*.

2. Let $p > 1$ be a prime number. It can be shown that Z_p^* is a cyclic group, that is there exists a generator $g \in Z_p^*$ such that $Z_p^* = \{g^1, \ldots, g^(p-1)\}$. For $y \in Z*_p$ we let $\log_g(y)$ to denote the smallest non-negative integer i for which $g^i = y$. For example $\log_g(1) = 0$ and $\log_g(g) = 1$. (Note that $0 \geq \log_g(y) \geq p-1$.) Show that y is a QR in $Z*_p$ if and only $\log_g(y)$ is an even number.

Exercise 3 (20 points)**.** We can define *chosen ciphertext security* for public key encryption schemes in the same way that we defined them for private key encryption schemes: the adversary gets access to a decryption box before the challenge and after receiving the challenge ciphertext y^* is allowed

to query the box on every string y *except* for y^*. The only difference is that the adversary gets initially the encryption key as another input. As usual we say the scheme is secure against Chosen Ciphertext Attack (CCA secure for short) if no poly-time adversary can win with $1/2 + \epsilon(n)$ probability where ϵ is poly-bounded.

1. Show that every public key encryption that is built from a trapdoor permutation family as in Exercise 1 (when we encrypt an n bit message by encrypting each bit individually) is *not* CCA secure. See footnote for hint[1]

2. Show that the specific encryption scheme based on Rabin's trapdoor permutation family has an even more devastating attack: show that given access to a decryption box for this scheme a polynomial-time adversary can recover the *private key* with high probability using only a constant number of queries.

Exercise 4 (20 points)**.** In this question we complete and formalize the proof of the Chinese Reminder Theorem.

1. Let G_1, G_2 be abelian groups where $+_i$ is the group operation of G_i. We define the *direct product* $G \stackrel{\text{def}}{=} G_1 \times G_2$, which consists of all pairs (a_1, a_2) where $a_i \in G_i$. We can view G in a natural way as an abelian group if we define the group operation $+_G$ component-wise: $(a_1, a_2) +_G (b_1, b_2) = (a_1 +_1 b_1, a_2 +_k b_2)$. Prove that G is indeed an abelian group with respect to $+_G$.

2. A *group homomorphism* is a function f from an abelian group G to an abelian group H that preserves the group operation; i.e., $f(a) +_H f(b) = f(a +_G b)$ for all $a, b \in G$. Let $n = p \cdot q$ where $\gcd(p, q) = 1$. Let f be a mapping from Z_n to $Z_p \times Z_q$ such that $f(x) = (x \mod p, x \mod q)$. Show that f is a group homomorphism.

3. Show that $f(x) \in Z_p^* \times Z_q^*$ if and only if $x \in Z_n^*$.

4. By the previous question, f can be viewed as a mapping from Z_n^* to $Z_p^* \times Z_q^*$. Show that in this case f is also a group homomorphism.

5. Give a polynomial-time algorithm to invert f (assuming that p, q are known). That is, give a poly$(\log(p) + \log(q))$-time algorithm that on input p, q and $(x', x'') \in Z_p \times Z_q$ outputs $x \in \{1..pq\}$ such that $x' = x \pmod p$ and $x'' = x \pmod q$. See footnote for hint.[2]

6. Prove that $|Z_n^*| = (p-1) \cdot (q-1)$, and conclude that f is an isomorphism from Z_n^* to $Z_p^* \times Z_q^*$ as well as from Z_n to $Z_p \times Z_q$.

Exercise 5 (20 points)**.** Suppose that the RSA Assumption fails "somewhat" on a particular composite number N and e with $gcd(e, \varphi(N)) = 1$, in the sense that there is a T-time algorithm A such that

$$\Pr_{y \leftarrow_R \mathbb{Z}_N^*}[A(y) = x \text{ s.t. } x^e = y \pmod N] > 0.01$$

Show that there is a $100(\log N)^{100} \cdot T$-time algorithm B that breaks the RSA Assumption completely for N, e in the sense that

$$\Pr_{y \leftarrow_R \mathbb{Z}_N^*}[A(y) = x \text{ s.t. } x^e = y \pmod N] > 0.99$$

[1] **Hint:** Show that every encryption scheme that works in a bit-by-bit fashion is not CCA secure.

[2] **Hint:** Use the gcd algorithm to find integers α, β such that $\alpha p + \beta q = 1$. Use this to invert f on the inputs $(1, 0)$ and $(0, 1)$ and proceed using linearity.

See footnote for hint[3]

Exercise 6 (20 bonus points + 10 extra bonus points)**.** Prove that the following algorithm outputs a random number R in $\{1..N\}$ together with R's factorization:

1. Generate a random decreasing sequence $N \geq S_1 \geq \cdots \geq S_\ell = 1$, by choosing S_1 at random in $\{1..N\}$, S_2 at random in $\{1..S_1\}$ and so on until reaching 1.
2. Let $(P_1, \ldots, P_{\ell'})$ denote the S_i's in this sequence that are *prime*, and let $R = P_1 \cdots P_{\ell'}$. If $R \leq N$ then with probability R/N output R together with its factorization $(P_1, \ldots, P_{\ell'})$.
3. If we did not output a number in Step 2, go back to Step 1.

You need to prove that (a) conditioned on outputting a number R, R will be distributed uniformly in $\{1..N\}$ and (b) that the algorithm runs in time poly$(\log N)$. Both follow by showing that for every $R \in \{1..N\}$, the probability that R is output in one iteration of the algorithm in Step 2 is $(1/N)\prod_{P \leq N}(1 - 1/P)$ (note that this product is over all $P \leq N$, and not just P that divides N). You can use the fact that $\prod_{P \leq N}(1 - 1/P) \geq \Omega(1/\log N)$ (or for 10 extra points prove it by using the method from the lecture notes to bound $\sum_{P \leq N} \log P$). See footnote for hint[4]

The algorithm of this exercise is due to Adam Kalai, although please do not look at his paper until after you have completed the exercise on your own.

BTW a clarification - there is really no difference between bonus points and "plain" points. I designate some questions as bonus when they are harder or a bit tangential to the course. I also always make sure that you can earn 100 points for the home work without doing the bonus questions. Thus I suggest that you work on the other questions first, and tackle the bonus questions later.

[3] **Hint:** Use the fact that $y^{1/e}r = (yr)^{1/e} \pmod N$ for every $r \in \mathbb{Z}_N^*$.

[4] **Hint:** Think of the random number S_1 as chosen as follows: we output N with probability $1/N$, otherwise we output $N-1$ with probability $1/(N-1)$, otherwise we output $N-2$ with probability $1/(N-2)$, etc..