Homework 6

1 Problem 1 (15 points)

(a) Show that the original version of the decisional Diffie Hellman problem that we saw in class is easy. That is, fix a prime p. You are given

$$(g, g^a \bmod p, g^b \bmod p, h)$$

where g is a random generator of $\mathbb{Z}_p^*$, $a, b \leftarrow \mathbb{Z}_{p-1}$, and h is either $g^{ab} \bmod p$ or $g^c \bmod p$ for a random $c \in \mathbb{Z}_{p-1}$.

Show how to tell whether $h = g^c \bmod p$ or $h = g^{ab} \bmod p$.

(b) Explain why, despite the above attack, the *computational* Diffie Hellman problem might still be hard

(c) Generalize the above attack as follows. Suppose $\mathbb{G}$ is a cyclic finite group of order N, and suppose N has a small factor r. Show that the decisional Diffie Hellman problem can be broken in time proportional to r (and polylogarithmic in N).

(d) A number N is t-smooth if all of its prime factors are at most t. Let $\mathbb{G}$ be a cyclic finite group of order N, where N is the product of distinct prime factors and N is t-smooth for some small t. Show that the discrete log problem is easy in $\mathbb{G}$: given any g and g^a, it is possible efficiently recover a, with a running time that grows with t, but is otherwise logarithmic in N. The Chinese Remainder Theorem will be helpful here.

(e) Show that the discrete log problem is easy over $\mathbb{Z}_N^*$ for any smooth N. That is, if N is t-smooth, you should give an algorithm for the discrete log over $\mathbb{Z}_N^*$ whose running time grows with t, but is otherwise logarithmic in N

Note that the N in part (e) is different from the N in part (d). In part (d), N is the order of the group (the number such that $g^N = 1$), whereas in (e), the order of the group is something very different.

2 Problem 2 (10 points)

Consider the following commitment scheme, built from a group GrGen:

- $\mathsf{Setup}()$: run $(\mathbb{G}, g, p) \leftarrow \mathsf{GrGen}(\lambda)$. We will assume GrGen always produces a prime p. Choose a random $a \in \mathbb{Z}_p$, and compute $h = g^a \in \mathbb{G}$. The commitment key is $k = (g, h)$.
- $\mathsf{Com}((g, h), m; r)$: We will assume the message space is $\mathbb{Z}_p$. Output $g^m h^r$, where r is a random element in $\mathbb{Z}_p$.

(a) Show that the scheme is perfectly hiding.

(b) Show that the scheme is computationally binding, assuming the discrete log problem is hard for $\mathbb{G}$.

3 Problem 3 (16 points)

Let $N = pq$ be the product of two primes. In this problem, we will see that, in addition to p and q being large, it is important that $p-1$ and $q-1$ have large prime factors.

(a) Suppose you know an integer r that is a multiple of $p-1$, but not $q-1$. Explain how to factor N.

(b) Suppose $p-1$ is t-smooth (recall that this means all of the factors of $p-1$ are at most t). Explain how to compute an integer r that is a multiple of $p-1$. Your r should be no larger than about p^t (so its bit length is at most $t \log_2 p$), and should take time polynomial in t and $\log_2 p$ to compute.

(c) You are not quite done, as your multiple r might also be a multiple of $q-1$. Explain how to detect this case.

(d) If your r is a multiple of both $p-1$ and $q-1$, then show how to derive a different integer r' that is a multiple of $p-1$ but not $q-1$, or vice versa. Assume $p \neq q$ (if $p = q$, we can easily factor by taking square roots).

One option to avoid this attack is to choose p, q to be safe primes, which means that $(p-1)/2$ and $(q-1)/2$ are also prime. However, this is not actually necessary, as it turns out that a random large prime p will, with high probability, have $p-1$ not be smooth.

4 Problem 4 (9 points)

In this problem, we will see how to combine cryptosystems, so that the resulting scheme is secure, as long as *either* component is secure.

(a) Let $G_0, G_1 : \{0,1\}^\lambda \to \{0,1\}^{4\lambda}$ be two efficiently computable functions. Suppose you know that either G_0 or G_1 is a secure pseudorandom generator, but you do not know which one. Construct a new function G that is guaranteed to be a secure pseudorandom generator.

(b) Let $(\mathsf{Gen}_0, F_0, F_0^{-1})$ and $(\mathsf{Gen}_1, F_1, F_1^{-1})$ be two trapdoor permutations with the same domain/range. Suppose you know that one of them is secure, but you do not know which one. The insecure scheme is at least guaranteed to be correct. Construct a new trapdoor permutation $(\mathsf{Gen}, F, F^{-1})$ that is guaranteed to be secure.

(c) Let $(\mathsf{Enc}_0, \mathsf{Dec}_0)$ and $(\mathsf{Enc}_1, \mathsf{Dec}_1)$ be two (secret key) encryption schemes, both with finite message space $\mathcal{M} = \{0,1\}^n$ and finite ciphertext space $\mathcal{C} = \{0,1\}^{m(\lambda)}$ where $m(\lambda) > n$. Suppose you know that one of them is CPA secure, but you do not know which one. The insecure scheme is at least guaranteed to be correct. Construct a new encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$, also for message space $\mathcal{M}$, that is guaranteed to be CPA secure. The scheme may have a ciphertext space that is different than $\mathcal{C}$.

5 BONUS Problem 5 (5 points)

For problem 4(c), suppose that the encryption schemes $(\mathsf{Enc}_0, \mathsf{Dec}_0)$ and $(\mathsf{Enc}_1, \mathsf{Dec}_1)$ worked for arbitrary-length messages, meaning the message space and ciphertext space were $\{0,1\}^*$. Explain why your solution to 4(c) will not work in this setting. Devise a new way to combine $(\mathsf{Enc}_0, \mathsf{Dec}_0)$ and $(\mathsf{Enc}_1, \mathsf{Dec}_1)$ that will be secure as long as at least one of the two schemes is secure.