6.875/18.425J Cryptography and Cryptanalysis March 13, 2005

Handout 9: Problem Set #4

This problem set is due on: March 30, 2005.

Problem 1 - PRG $\Rightarrow$ OWF

Prove that the existence of a secure Pseudo-Random Generator implies the existence of a length-preserving One-Way Function

Problem 2 - PRGs and Permutations

Let G be a pseudorandom generator with expansion function $\ell(k)$, and let h be any length-preserving permutation (which is not necessarily polynomial-time computable).

A: Is it necessarily true that the distribution $h(G(s))$ (where s is chosen uniformly at random from $\{0,1\}k$) is indistinguishable from the uniform distribution over $\{0,1\}\ell(k)$? Is $h(G(s))$ a pseudorandom generator? Justify your answers.

B: Is it necessarily true that the distribution $G(h(s))$ (where s is chosen uniformly at random from $\{0,1\}k$) is indistinguishable from the uniform distribution over $\{0,1\}\ell(k)$? Is $G(h(s))$ a pseudorandom generator? Justify your answers.

C: Will your answers to the previous parts change if it is known that h is polynomial-time computable?

Problem 3 - Composing PRGs

Let G_1, G_2 be PRGs with expansion functions $\ell_1(k)$,$\ell_2(k)$ (respectively). For each of the candidates below, justify whether the function is a PRG or not. If yes, then provide a security reduction. If not, provide a counterexample.

A: $G_A(x) = \text{reverse}(G_1(x))$ where the reverse() reverses the bits of its argument.

B: $G_B(x) = G_1(x) \circ G_2(x)$

C: $G_C(x \circ y) = G_1(x) \circ G_2(y)$, where $|x| = |y|$ or $|x| = |y| + 1$

D: $G_D(x) = G_2(G_1(x))$

E: $G_E(x) = G_1(x) \oplus (x \circ 0^{\ell_1(|x|)-|x|})$

Problem 4 - Unpredictability ⇒ Indistinguishability

In class we proved that if the output of a generator $G : \{0,1\}^k \to \{0,1\}^n$ (here n is some polynomial of k) passes the next bit unpredictability test, then it passes all statistical tests. The proof used a hybrid argument to show that if there was a polynomial time statistical test A that distinguishes a completely random string from one generated by G, then the test could distinguish between a string in which the first i bits are from G and the rest random, and a string in which the first $i+1$ bits are from G and the rest random. To complete the proof, we then need to show how to use this to predict the next bit $(i+1)$ from the first i bits with probability non-negligible better than $\frac{1}{2}$. Below are some suggestions of how to produce such a guess for the $(i+1)$st bit.

For each of the suggested predictors, give a convincing explanation of whether it is indeed a good predictor or not. Supply a formal proof for *one* of the good predictors. That is, prove that it indeed guesses correctly with probability better than $\frac{1}{2} + \frac{1}{Q(k)}$ for some polynomial Q. Denote by G_m the first m bits of $G(x)$ (where x is a random seed), and by R_m (or R'_m) a sequence of m random bit chosen from the uniform distribution. Assume without loss of generality that $\Pr[A(G_i R_{n-i}) = 0] = p$, and that $\Pr[A(G_{i+1} R_{n-i-1}) = 0] = p + \frac{1}{k^c}$ for some $c > 0$ (that is, we are assuming w.l.o.g. that A outputs 0 more often when the $(i+1)$st bit is from G). We are now given i bits G_i, and want to guess the next bit. Consider the following predictors.

(a) Run the test A first on $G_i 0 R_{n-i-1}$ and call the output a_0. Then run A on $G_i 1 R'_{n-i-1}$ and call the output a_1. If $a_0 = a_1$ output 0, otherwise output 1.

(b) Run the test A first on $G_i 0 R_{n-i-1}$ and call the output a_0. Then run A on $G_i 1 R'_{n-i-1}$ and call the output a_1. If $a_0 = a_1$ choose the output to be 0 or 1 randomly (with probability $\frac{1}{2}$). Otherwise, output the bit b for which $a_b = 0$ (that is, if $a_0 = 0$ output 0, and if $a_1 = 0$ output 1).

(c) Run the test A on $G_i R_{n-i}$. If the answer is 0, output the first bit of R_{n-i} (which is the $(i+1)$st bit in the string above). If the answer is 1, output the negation of that bit.

(d) Run the test A on $G_i 0 R_{n-i-1}$ for polynomially many times (each time with new independent R_{n-i-1}), and count how many times A outputs 0. If this fraction is closer to $p + \frac{1}{k^c}$ than to p, then output 0, otherwise output 1.