Introduction to Cryptography
Courant, Fall 2016

Homework 3

Instructor: Oded Regev
Student: YOUR NAME HERE

Homework is due by **7am of Oct 3**. Send by email to both "regev" (under the cs.nyu.edu domain) and "avt237@nyu.edu" with subject line "CSCI-GA 3210 Homework 3" and name the attachment "YOUR NAME HERE HW3.tex/pdf". There is no need to print it. Start early!

Instructions. Solutions must be typeset in LaTeX (a template for this homework is available on the course web page). Your work will be graded on *correctness*, *clarity*, and *conciseness*. You should only submit work that you believe to be correct; if you cannot solve a problem completely, you will get significantly more partial credit if you clearly identify the gap(s) in your solution. It is good practice to start any long solution with an informal (but accurate) "proof summary" that describes the main idea.

You are expected to read all the hints either before or after submission, but before the next class.

You may collaborate with others on this problem set and consult external sources. However, you must ***write your own solutions***. You must also ***list your collaborators/sources*** for each problem.

1. [1] A *collection of one-way functions* is a family $F = \{f_s : D_s \to R_s\}_{s \in S}$ satisfying:

   1. Easy to sample function: there exists a PPT algorithm Gen that outputs some $s \in S$ (according to some distribution);
   2. Easy to sample from domain: there exists a PPT algorithm D such that $D(s)$ outputs some $x \in D_s$ (according to some distribution);
   3. Easy to evaluate: there exists a PPT algorithm F such that for all $s \in S, x \in D_s$ we have $F(s, x) = f_s(x)$;
   4. Hard to invert: for any (non-uniform) PPT I,

   $$\Pr_{s \leftarrow S(1^n),\, x \leftarrow D(s)}[I(s, f_s(x)) \in f_s^{-1}(f_s(x))] = \text{negl}(n) .$$

   (The only significant change here compared to the definition of OWF is the introduction of s.) In this question we prove that there exists a collection $\{f_s\}$ of one-way functions if and only if there exists a one-way function f.

   (a) (2 points) Prove the "if" part.

   Solution: Suppose we have a one-way function $f(x)$. Clearly if we define a family of functions where each member is the original function f(x), then these functions are all easy to compute and hard to invert by virtue of the fact that f(x) is itself one-way. So it only remains to show that we can define an easy way to sample a function, and an easy way to sample from the domain. To do this, we can define our $Gen()$ function as $Gen(1^n) = n$, and our sampling function as drawing a random string from $0, 1^n$. Both of these can be done efficiently, and thus we have defined a (somewhat trivial) family of functions without knowing anything about the specifc form of f.

   (b) (3 points) Prove the "only if" part. We recommend you make the simplifying assumption that the set of keys S is $\{0, 1\}^n$ with the uniform distribution and also that the domain of all the functions in the collection is $\{0, 1\}^n$, again with the uniform distribution. So the collection of OWFs is $\{f_s : \{0, 1\}^n \to \{0, 1\}^n\}_{s \in \{0,1\}^n}$ and we are given just one deterministic algorithm F that takes a key $s \in \{0, 1\}^n$ and an input $x \in \{0, 1\}^n$ and outputs $f_s(x)$ (there is no need anymore for Gen

[1] A question from Peikert's class

and D). Once you are done with this, you can try to extend it to the general setting (but start your solution with the simpler case).

Solution: Suppose we have a family of functions. Note that it is not necessarily the case that each member function in the family is one-way (since the definition averages of the variability in the index function, there could be some non-one-way functions in the family such that the overall probability of inversion is nonetheless negligible). This means we cannot simply fix some i and then use $f_i(x)$ as our example. However, if we assume that the set of keys S is uniformly distributed on $0, 1^n$, Then we can construct the mapping $g : 0, 1^{2n} \to 0, 1^{2n}, g(s, x) = (s, f_s(x))$. Note that this function is essentially a wrapper function that combines the selection of the index and the evaluation of $f_s(x)$ in one step. Notably, the adversary still has full knowledge of the key s. In particular, we have that $Pr_{s \leftarrow S(1^n),\, x \leftarrow D(s)}[I(s, f_s(x)) \in f_s^{-1}(f_s(x))] = Pr_{x \in \{0,1\}^n}[I(1^n, g(s, x)) \in f^{-1}(g(s, x))] \in \text{negl}(n)$, Since in both cases we are now averaging over the index s, with the only difference being that the RHS includes variations in s as variations over the space of inputs to g, whereas the LHS includes variations in s by averaging over all members of the family of functions f_i. Therefore, $g(s, x) \in negl(n)$.

2. (2 points) *(Expanding a PRG.[♣])* Suggest a construction that we can use to show that the existence of a PRG with output length $\ell(n) = n + 1$ implies the existence of a PRG with any $\text{poly}(n)$ output length. If you feel adventurous, try to suggest a way to prove its correctness.

Solution: Suppose we take the output of the PRG $G(k)$ as use the first k bits of it as the input to G again, and repeatedly apply this procedure $poly(n) - k$ times. This construction based on G(x). which I will denote $G^{poly(n)-k}(x)$, clearly still runs in polynomial time, since G runs in polynomial time. Thus it remains to show that $G^{poly(n)-k}(x)$ is in fact a psuedorandom generator. This is not trivially true, because we are no longer using a random seed as the input to the algorithm, but rather a seed that is the output of a pseudorandom generator.

Let U_k be the uniform distribution over k bits. Then by the fact that G is a PRG, we know that the distributions U_{k+1} and $G(U_k)$ are indistringuishable, despite the fact that only haf of the strings in U_{k+1} are in the range of G. Furthermore, we assume without giving proof that if two distributions are indistinguishable, then the composition of any function with those distributions is also indistiguishable. A heuristic argument for that might be that although $G(U_k)$ may have a restricted domain relative to U_{k+1}, we are restricting our sample used in the calculation of statistical distance as well, so that the probabilities of any event in the range of $g^i(x)$ are still the same as they are in $G(x)$, and the statistical distance calculation remains unchanged. Now in order to show that $G^{poly(n)-k}(x)$ is pseudorandom, we need to show that for any PPT distinguisher D, there is a negligible function $negl(n) : |P[D(G^{poly(n)-k}(s) = 1] - P[D(u) = 1]| \leq negl(n)$. Note that by the triangle inequality, we can write $|P[D(G^{poly(n)-k}(s) = 1] - P[D(u) = 1]| \leq \sum_{i=k \to poly(n)} |P[D(G^{i-k}(s) = 1] - P[D(G^{i-k-1}(s) = 1]|$, where $(G^0(s) = u$ is simply our original uniform string. Since compostion of G with U_k does not change the underlying distribution, this sum is equal to $\sum_{i=k \to poly(n)} |P[D(G^{i-k}(s) = 1] - P[D(u) = 1]|$. so we know that each of

[♣]Another "food-for-thought" question; you are not required to solve it fully, but you are required to demonstrate that you thought about it seriously.

the terms in this sum is a negligible function of n, and therefore the resulting probability is bounded above by some negligible function.

3. (2 points) *(Constructing a PRG.♣)* Try to suggest ways to build a PRG from a OWF. For instance, say we take a one-way function $f : \{0,1\}^n \to \{0,1\}^n$. Explain why $g : \{0,1\}^n \to \{0,1\}^{2n}$ defined by $g(x) = (f(x), x)$ is not a PRG. How about $g(x) = (f(x), x_1)$, where x_1 is the first bit of x? Explain how taking f to be a one-way *permutation* helps a bit, but still does not give us a PRG. Suggest a way one can try to fix the problem.

Solution: The function defined by $g(x) = (f(x), x)$ described in the question is not a PRG, because the range of g(x) is only 2^{-n} fraction of all possible outputs, so we could easily construct a distringuisher that tests f on the second half of the string, and then outputs 1 whenever it matches the first half of the string. The probability of getting a 1 if the string is generated by g is 1, whereas the probability of getting a 1 from a uniform string is 2^{-n}. Thus $|P[D(G(s)) = 1] - P[D(u) = 1]| = 1 - 2^{-n}$, which is not a negligible function.

If we instead define $g(x) = (f(x), x_1)$, where x_1 is the first bit of x, then the above distinguisher does not work, since we do not know the full set of bits for x. It is true that having one bit narrows down the set of possible x's to test, yielding a boost in probability for the pseudorandomly generated number by a constant factor of 2, but this is not enough to make the difference non-negligible. Thus, the above distinguisher does not disprive the proposition that G is a PRG as long as we concatenate a constant (O(1)) number of bits.

However, there is still a problem: while $f(x)$ may be one-way it is not necessarily injective. If f is not injective, we may have some information about the range of f, without actually being able to invert it. Then we could always guess 1 whenever the input x is not in the range of f. This would happen with probability 0 whenever x is pseudorandomly generated, and with probability 1 whenever x is randomly generated. If the domain of f is sufficiently restricted, then this leaves open the possibility of distinguishing the two distributions with non-negligible probability. If the function f is injective however, then this distinguisher also fails to give an advantage, since our distinguished would output 1 in all cases with probability 1.

We still do not quite have a PRG however, as there is one more edge-case to consider. The fact that f is one-way does not imply that infromation is not leaked about certain bits, and in particular about x_1. Therefore we need to choose a bit that is adequately hidden by the applicaiton of f. Reading ahead in the book, it appears that these are called hard-core predicates). Aniother possible solution might be to use the XOR'ed value of all of the bits of x as the extra bit.

4. (2 points) Prove that there is no "statistical PRG", i.e., a (deterministic) function $g : \{0,1\}^n \to \{0,1\}^{\ell(n)}$ for some $\ell(n) > n$ such that $g(U_n)$ is within negligible total variation distance (also known as statistical distance) of $U_{\ell(n)}$.

Solution: We know that at most 2^n of the output strings in $0, 1^{\ell(n)}$ are in the range of g. Since g(x) is a PRG, we know that the two distributions are indistinguishable for all x in the range, whereas the probability for any value not in the range of g(x) is 0 for $g(U_n)$, and 2^{-n} for U_m. We can write the statistical distance as $\Delta(U_n, g(U_n)) = \frac{1}{2}\sum_{x \in g(X)} |U_n(x) - g(U_n(x))| + \frac{1}{2}\sum_{x \notin g(X)} |U_n(x) - g(U_n(x))|$. Recall that there is an alternative formulation of the statistical distance that we can apply to each of the subsets above: $\Delta(U_n, g(U_n)) = \sup_{A \subseteq g(X)} |U_n(A) - g(U_n(A))| + \sup_{A \not\subseteq g(X)} |U_n(A) - g(U_n(A))|$

Note that the subset A which maximizes the the terms above can be thought of as a distinguisher function, where the $D(x) = 1 \forall x \in A$, and 0 otherwise. In particular, for the term on the right, $U_n(x) = 2^{-\ell(n)}$, and $g(U_n(x)) = 0 \forall x \notin g(X)$, so the value of this term is maximized when A is the entire set of $x : x \notin g(X)$. Since this set has cardinality $2^{\ell(n)} - 2^n$, the value of the term on the right is $1 - 2^{n-\ell(n)}$. Now for the left term, we can imagine the subset A which maximizes this sum as some distinguisher D which outputs 1 if x is in A, i.e. $\sup_{A \not\subseteq g(X)} |U_n(A) - g(U_n(A))| = |P[D(G(s)) = 1] - P[D(u) = 1]|$. Though we may not know the exact form of D, we know that since g(x) is pseudorandom, $|P[D(G(s)) = 1] - P[D(u) = 1] \in negl(n)$. Therefore $\Delta(U_n, g(U_n)) = 1 - 2^{n-\ell(n)} + negl(n)$, which is not a negligible function as long as $\ell(n) > n$.