ICS 180: Introduction to Cryptography 6/14/2004

Solutions to homework 5

1 Constructing a PRG from a PRF

This question is designed so that you see a relation between a PRF and a PRG. You have seen in class that with some work one can build a PRF out of any PRG. But PRF does seem like a more powerful construct, so the other direction, construction of a PRG from a PRF should be easy. But how shall this be done exactly?

Let$\{f_s \mid s \in \{0,1\}^\tau\}_{\tau=1,2,...}$ be a PRF family, where for each τ and each $s \in \{0,1\}^\tau$, function f_s maps domain $\{0,1\}^\tau$ onto the same range $\{0,1\}^\tau$. (Using the notation from the lecture and the notes, we'd say that $l(\tau) = L(\tau) = \tau$.)

Consider the following attempts to construct a PRG from this PRF family. For each of the attempts, either prove that the PRG is secure or prove that it is not, by showing an efficient algorithm that distinguishes its outputs from random strings:

1. $G_1(x) = [f_x(0^\tau) \mid f_x(1^\tau)]$ for $x \in \{0,1\}^\tau$
2. $G_2(x) = [f_{0^\tau}(x) \mid f_{1^\tau}(x)]$ for $x \in \{0,1\}^\tau$

Note that both constructions, on purpose, are done in a way so that the G_i's are trivially stretching: $|G_i(x)| = 2|x|$ for both $i = 1, 2$.

Hint: First, recall what a (secure) PRG is and what a (secure) PRF is. If you want to prove that a PRG construction is *secure*, use one of the two security arguments we have had. Namely, either prove that some two required probability distribution are indistinguishable directly by a series of transformations (for example as in the solutions to problem (1.1) in homework 4). Or, prove it by contradiction, i.e. assume that there exists a PPT adversary A that breaks the PRG security property for the construction G_1 or G_2, and use that adversary to create a PPT attack A' that breaks the PRF security property for the function family $\{f_s\}$.

If you want to show that the PRG construction is *insecure*, you can do so similarly as in the problem (1.2) in homework 4, i.e. by showing that for *some* PRF family $\{f_s\}$, the family itself is a secure PRF family, but the G_i construction (for i either 1 or 2) fails to produce a pseudorandom number generator. How can you do this? Recall the method we used in problem (1.2) of homework 4 and apply it in this case. Namely, try to *create* function family $\{f_s\}$ from any PRF family $\{\bar{f}_s\}$ s.t. $\{f_s\}$ remains a PRF family, but it makes the G_i construction fail as a PRG.

Solution:

1.1

The G_1 construction actually does make a secure PRG. G_1 is a good PRG if the following probability distributions are indistinguishable:

$$\{G_1(s)\}_{s \leftarrow \{0,1\}^\tau} \approx \{r\}_{r \leftarrow \{0,1\}^{2\tau}}$$

I claim that this is indeed the case. One way to argue this is the following: Since $\{f_s\}$ is a PRF, for every (efficient) adversary A we have

$$\{A^{f_s}(1^\tau)\}_{s\leftarrow\{0,1\}^\tau} \approx \{A^R(1^\tau)\}_{R\leftarrow RNDFCT(\tau,\tau)} \quad (1)$$

which reads: "the distribution of outputs of A on input 1^τ and on access to function f_s, where s is a random τ-bit seed is indistinguishable from the distribution of outputs of A on input 1^τ and on access to a random function R".

An intuitive way to say this is that for every adversary A, A's conversation with f_s (for a randomly chosen s) *looks the same* (i.e. are indistinguishable) as A's conversation with a random function R. Therefore, in particular, if we take an algorithm A which asks only two queries to the function F is has access to: $x_1 = 0^\tau$ and $x_2 = 1^\tau$, then this A cannot tell the difference between the returned values $(F(x_1), F(x_2)) = (f_s(x_1), f_s(x_2))$ for random s, and values $(R(x_1), R(x_2))$ for a random function R. In other words, it must be that:

$$\{(f_s(x_1), f_s(x_2))\}_{s\leftarrow\{0,1\}^\tau} \approx \{(R(x_1), R(x_2))\}_{R\leftarrow RNDFCT(\tau,\tau)}$$

But clearly we have

$$\{(R(x_1), R(x_2))\}_{R\leftarrow RNDFCT(\tau,\tau)} = \{(r_1, r_2)\}_{r_1,r_2\leftarrow\{0,1\}^\tau}$$

And therefore we get what we wanted, i.e. that

$$\{G_1(s)\}_{s\leftarrow\{0,1\}^\tau} = \{(f_s(x_1), f_s(x_2))\}_{s\leftarrow\{0,1\}^\tau} \approx \{(r_1, r_2)\}_{r_1,r_2\leftarrow\{0,1\}^\tau} = \{[r_1|r_2]\}_{r_1,r_1\leftarrow\{0,1\}^\tau}$$

1.2

The G_2 construction is insecure in general. I.e., if PRFs exist at all then you can have a PRF which clearly fails to make the G_2 construction a PRG. The reason is that the G_2 construction attempts to use the PRF construction $\{f_s\}$ by taking *particular two instances* f_{s_1} and f_{s_2}, where $s_1 = 0^\tau$ and $s_1 = 1^\tau$ of the PRF family $\{f_s\}$. Now, when we say that $\{f_s\}$ is a PRF family we claim that the algorithm which defines this function (remember that a PRF must be efficiently computable) has some nice random-function-like properties but only if it is instantiated with a *random seed* $s \leftarrow \{0,1\}^\tau$. These "nice random-function-like" properties are expressed in equation (1) above, and you can see that the statement is made for a *random* s. But the G_2 construction attempts to use the $\{f_s\}$ PRF family not by picking index s at random but by looking at the particular instances f_{s_1}, f_{s_2}. This is like running a secure encryption algorithm not on a random key but on some two particular fixed keys k_1 and k_2. Nobody says that a good cipher must behave nicely on *every* key. Quite the opposite: Some keys can indeed be bad. As long as there are few of them, the chances of picking a bad one are negligible and therefore such "weakness" does not matter.

We will see why this is indeed true for PRFs: Assuming that PRFs exist at all, take any PRF family $\{f_s\}$ and modify it as follows. Define a new function family $\{f'_s\}$ by assigning $f'_s(x) = f_s(x)$ for all x and all s except for $s = s_1$ or $s = s_1$ in which case define $f'_{s_1}(x) = 0$ and $f'_{s_2}(x) = 0$ for all x. In other words, we take a good PRF and we artificially weaken it by fixing two instances of it, for indeces s_1 and s_2, to be a constant function, which always return 0. This transformation weakened the original PRF slightly, but in fact it did *not* make the modified function insecure. The reason is that:

$$\{A^{f_s}(1^\tau)\}_{s\leftarrow\{0,1\}^\tau} \approx \{A^{f'_s}(1^\tau)\}_{s\leftarrow\{0,1\}^\tau}$$

I.e. that conversations with a random instance of f'_s are indistinguishable from conversations with a random instance of f_s. Why is that? Because the only chance for A to distinguish f_s from f'_s is when s happens to be either s_1 or s_2. But since s is chosen at random from a set of 2^τ strings, that happens with probability only $2/2^\tau \leq negl(\tau)$.

And therefore, by transitivity of indistinguishability we have that if $\{f_s\}$ is a PRF then so is $\{f'_s\}$, i.e. if

$$\{A^{f_s}(1^\tau)\}_{s\leftarrow\{0,1\}^\tau} \approx \{A^R(1^\tau)\}_{R\leftarrow RNDFCT(\tau,\tau)}$$

then

$$\{A^{f'_s}(1^\tau)\}_{s\leftarrow\{0,1\}^\tau} \approx \{A^R(1^\tau)\}_{R\leftarrow RNDFCT(\tau,\tau)}$$

Now, the new PRF $\{f'_s\}$ clearly breaks the G_2 construction because

$$\{G_2(x)\}_{x\leftarrow\{0,1\}^\tau} = \{(f_{s_1}(x), f_{s_2}(x))\}_{x\leftarrow\{0,1\}^\tau} = \{(0,0)\}$$

In other words, G_2 always returns a constant!

□

2 Extending the range of a PRF

Let $\{f_s\}$ be a PRF family as above. Below there are several attempts to make another PRF family $\{f'_s\}$, using the existing PRF family $\{f_s\}$, s.t. for each s, $f'_s : \{0,1\}^\tau \rightarrow \{0,1\}^{2\tau}$. In other words, the outputs of f' are twice longer than the outputs of the existing PRF family.[1]

In each case either prove that the result is also a secure PRF or show a PPT algorithm which breaks it, i.e. which distinguishes between conversations with f' and conversations with a truly random function.

1. $f'_s(x) = [f_s(0^\tau) \mid f_s(x)]$
2. $f'_s(x) = [f_{0^\tau}(x) \mid f_s(x)]$
3. $f'_s(x) = [f_s(x) \mid f_s(\bar{x})]$ ($\bar{x}$ is a bitwise negation of x)
4. $f'_s(x) = [f_s(0x) \mid f_s(1x)]$

Hint: The methodology you can use here differs slightly from the one used in question (1): Namely, if a construction fails then you can actually show an attack algorithm A which distinguishes conversations with f'_s from conversations with a truly random function $R : \{0,1\}^\tau \rightarrow \{0,1\}^{2\tau}$, such that this attack will work regardless of what the underlying PRF family $\{f_s\}$ is. In other words, I don't think you'll need to show a special PRF family $\{f_s\}$ which fails the f'_s construction: The wrong construction can be broken for *every* PRF family $\{f_s\}$. However, if you think that you one of the constructions is secure, you should use one fo the standard techniques to show that, as in question (1).

[1] This type of question is a common crypto issue which we have seen with regards to pseudorandom generators [PRG] and to encryption schemes [EΣ]. The question is: Given an algorithm which provides a EΣ/PRG/PRF functionality to some degree, e.g. a PRG which stretches its inputs by just one bit, or an EΣ which encrypts only one-bit long messages, or as we have here a PRF which maps $\{0,1\}^\tau$ onto $\{0,1\}^\tau$), can you use this algorithm as a black box to provide the PRG/EΣ/PRF functionality with better parameters, i.e. a PRG which stretches its inputs by any polynomial number of bits, or an EΣ which encrypts messages of any (polynomial) length, or as here, a PRF which produces twice longer outputs.

Solution:

2.1

Is $f'_s(x) = [f_s(0^\tau) \mid f_s(x)]$ a good PRF?

No, it is insecure. Here is one attack A which on access to function F distinguishes if this function is f'_s for some random s or if it is a random function R. A can for example ask F on $x = 0^\tau$. In this case f'_s would return $f'_s(0^\tau) = [f_s(0^\tau) \mid f_s(0^\tau)]$, which, regardless of what f_s is, will always be a concatenation of two identifical strings. Therefore an attacker A can ask for $[y_L \mid y_R] = F(0^\tau)$ and output 1 if $y_L = y_R$. If $F = f'_s$ then $y_L = y_R = f_s(0^\tau)$, and so A will always output 1 in this case. On the other hand, if F is a random function then the probability that $y_L = y_R$ is only $1/2^\tau$, so

$$\{A^{f'_s}(1^\tau)\}_{s \leftarrow \{0,1\}^\tau} \not\approx \{A^R(1^\tau)\}_{R \leftarrow RNDFCT(\tau,\tau)}$$

There were other attacks that some of you pointed out: For example, once A makes any two queries to f'_s, it sees that the first half of the output is always the same tring.

2.2

Is $f'_s(x) = [f_{0^\tau}(x) \mid f_s(x)]$ a good PRF?

This one is also insecure. This tells you once more, similarly to what you've seen in problem 1.2, that you if something works for a random seed (or a random key, or random message, or a random whatever), it does not mean that it will work well for a particular fixed seed (or a key or a message, etc). Here the adversary can immediately recognize this new $\{f'_s\}$ construction from a random function by asking for $[y_L \mid Y_R] = f'_s(x)$ value on any query x, and then outputing 1 (meaning that A judges that he is talking to an instance of f'_s rather than a random function) if $y_L = f_{0^\tau}(x)$. Note that the adversary knows the description of the PRF family $\{f_s\}$ so he can apply the f_s algorithm on seed $s = 0^\tau$ and input x himself. Clearly, the chances that the left half of the random 2τ-bit value returned by a random function on query x is equal to $f_{0^\tau}(x)$ as well is only $1/2^\tau$. Hence again we have

$$\{A^{f'_s}(1^\tau)\}_{s \leftarrow \{0,1\}^\tau} \not\approx \{A^R(1^\tau)\}_{R \leftarrow RNDFCT(\tau,\tau)}$$

Another way to argue that this is an insecure construction could be by a following a very similar reasoning to the one of problem (1.2). Namely, as in problem (1.2) we can convince ourselves that this construction fails in general because for some PRF $\{f_s\}$, the particular index $s_1 = 0^\tau$ can behave arbitraily badly. In fact, as we argued in the solution to (1.2) above, f_{s_1} can be a constant function $f_{s_1}(x) = 0$ for all x and the $\{f_s\}$ function *family* can still be a PRF family. Of course, if $\{f_s\}$ is just such a family, with f_{0^τ} being a constant function returning 0 for every x, then the $\{f'_s\}$ construction is clearly wrong. An efficient A will notice after two queries to f'_s that the left halves of each output are the same.

2.3

Is $f'_s(x) = [f_s(x) \mid f_s(\bar{x})]$, where $\bar{x}$ is a bitwise negation of x, a secure PRF?

No, it's not. One attack could be for A to ask F first for $y = F(x)$ and then for $y' = F(\bar{x})$. If F is instantiated as f'_s for some s then y, y' have the form $y = [y_L \mid y_R]$ and

$y' = [y_R \mid y_L]$. This will always be the case for $F = f_s$ while for a random function, this behaviour happens with probability only $1/2^{2\tau}$.

2.4

How about $f'_s(x) = [f_s(0x) \mid f_s(1x)]$. Is it a PRF?

The last one is, for a change, a good construction. When you start attacking it you notice that you cannot get any repetitive patterns in the answers because whatever attacker's chosen x_i arguments are, he sees pairs of values of f_s on $x'_i = [0 \mid x_i]$ and $x''_i = [1 \mid x_i]$, and these x'_i and x''_i values are all different from one another. OK, but this is just an intuition that things are looking good, not a proof...

The proof is this: Assume $\{f'_s\}$ was not a PRF. Then there would exist an efficient A' s.t.

$$\{A'^{f'_s}(1^\tau)\}_{s \leftarrow \{0,1\}^\tau} \not\approx \{A'^{R'}(1^\tau)\}_{R' \leftarrow RNDFCT(\tau,2\tau)} \tag{2}$$

Let's use A' to construct an efficient break A against the pseudorandomness of the $\{f_s\}$ function family: A, on input 1^τ, and access to some function F, runs $A'(1^\tau)$, and every time A' makes a query x, A makes two queries to F, getting $y_L = F([0|x])$ and $y_R = F([1|x])$. A returns then to A' value $y = [y_L \mid y_R]$ as an answer to the query x of A'. When A' finally makes a judgment 0 or 1, A just repeats this judgement. Now, it's easy to see that equation (2) implies that

$$\{A^{f_s}(1^\tau)\}_{s \leftarrow \{0,1\}^\tau} \not\approx \{A^R(1^\tau)\}_{R \leftarrow RNDFCT(\tau,\tau)}$$

Why? Because if F is instantiated as f_s for a random s then A *simulated* to A' exactly the conversation that A' would have with function f'_s. On the other hand, if F was a random function $R : \{0,1\}^{\tau+1} \to \{0,1\}^\tau$ then the conversation that A' had was a conversation with function R' defined as $R'(x) = [R([0|x]) \mid R([1|x])]$. If R is a random function then so is R'. □