

Homework 6

Instructor: Anna Lysyanskaya

Due: March 7, 2007

Problem 1: PRG and symmetric-key authentication

In this problem, we show that a pseudorandom generator is basically all you need in symmetric-key cryptography.

- a. Suppose that Alice and Bob have agreed on a k -bit prime p . Suppose also that both of them are given secret a and b , $a \leftarrow \mathbb{Z}_p^*$, $b \leftarrow \mathbb{Z}_p^*$.

Alice wants to send to Bob a message $m \in \mathbb{Z}_p^*$. She does not mind sending it in the clear.

A malicious Eve can intercept m and replace it with m' . To safeguard against this behavior, Alice will use the secret a and b that she has agreed upon with Bob. She will use them as follows: she will compute a message authentication code (MAC) on m as follows: $c = am + b \pmod p$. She will then send (m, c) to Bob.

Eve may intercept (m, c) and replace it with (m', c') .

Upon receipt of (\hat{m}, \hat{c}) , Bob wants to verify that it originated with Alice. So he checks that $\hat{c} = a\hat{m} + b$. If the check passes, he accepts \hat{m} , otherwise he rejects.

Show that for any, even computationally unbounded adversary \mathcal{A} , for any p , and for any message $m \in \mathbb{Z}_p^*$

$$\Pr[a \leftarrow \mathbb{Z}_p^*; b \leftarrow \mathbb{Z}_p^*; c = am + b \pmod p; (m', c') \leftarrow \mathcal{A}(p, m, c) : m' \neq m \wedge am' + b = c'] = 1/p$$

Hint: Think of $f(m) = am + b \pmod p$ as a line. The adversary \mathcal{A} is given one point on this line, namely, $c = f(m)$. But you need two points to define a line, and so, from c alone, $f(m')$ could be any element of \mathbb{Z}_p^* whenever $m' \neq m$.

- b. Suppose that Alice and Bob have agreed upon a seed $s \leftarrow \{0, 1\}^k$ for a pseudorandom generator $G : \{0, 1\}^k \mapsto \{0, 1\}^{\ell(k)}$ where $\ell(k)$ is some large enough polynomial.

How can Alice use the pseudorandom generator in order to send to Bob a set of $\Theta(\ell(k)/k)$ authenticated k -bit messages? Give an algorithm for Alice that computes the message authentication code c_i for each message m_i . Give an algorithm for Bob that, on input (i, \hat{m}, \hat{c}) verifies that \hat{m} is indeed the i th message sent by Alice. Explain why Eve cannot substitute (i, m', c') for (i, m_i, c_i) such that $m' \neq m_i$ and yet (i, m', c') pass Bob's verification algorithm. (It is OK if the explanation is informal.)

- c. Describe how to modify your construction from part (b) so that (1) Alice can send each message m_i privately, i.e., so a computationally bounded Eve learns no information about it; and (2) Eve cannot modify m_i without Bob noticing (note that in the one-time pad cryptosystem, Eve can modify m even without knowing its value; for example, she can toggle any bit of m). Informally explain why it works.

Problem 2: Fun with one-way functions

In class, we saw an example of a one-way function f such that $f(f(x))$ is not a one-way function. We also saw that, if f is a one-way *permutation*, then so is $f(f(x))$. In this problem, we will look at other subtleties of this flavor.

- a. Let $f_1 : \{0, 1\}^k \mapsto \{0, 1\}^k$ and $f_2 : \{0, 1\}^k \mapsto \{0, 1\}^k$ be one-way functions. Let 'o' denote concatenation. It turns out that $f(x) = f_1(x) \circ f_2(x)$ is not necessarily a one-way function.

We can come up with contrived f_1 and f_2 , such that f_1 reveals half of the bits of x , and scrambles the other half (so it's impossible to invert), while f_2 scrambles the half that f_1 reveals, and reveals the half that f_1 scrambles. In other words, let $g : \{0, 1\}^{\lfloor k/2 \rfloor} \mapsto \{0, 1\}^{\lfloor k/2 \rfloor}$ be a one-way function. Let $f_1(x) = x_1 \circ g(x_2)$, and $f_2(x) = g(x'_1) \circ x'_2$, where $x = x_1 \circ x_2 = x'_1 \circ x'_2$, and $|x_2| = |x'_1| = \lfloor k/2 \rfloor$.

Explain why the resulting f is not a one-way function. Explain why the contrived f_1 and f_2 are one-way functions (here, you should sketch a reduction, but you don't need to formally analyze it, it is sufficient to just explain why it will work).

- b. The reason that the above is good to know is that, as we saw in lecture, the UNIX /etc/passwd file stores a one-way function of your password, instead of storing the password in the clear. So, if your password is x , it stores $y = f_1(x)$ for some one-way function f_1 . When you submit x logging in, it checks that $y = f_1(x)$. Now, if you use the same password on another machine, the other machine also stores some one-way function of your password, $f_2(x)$. Suddenly, it no longer follows that your password is protected!

But maybe your password is still protected as long as f_1 and f_2 are not too contrived? Maybe for some f_1 and f_2 , it can still be the case that $f(x) = f_1(x) \circ f_2(x)$ is a one-way function? Assuming that there exists a one-way function $g : \{0, 1\}^{\lfloor k/2 \rfloor} \mapsto \{0, 1\}^{\lfloor k/2 \rfloor}$, give a construction of f_1 and f_2 such that f defined above is a one-way function. Explain why f is a one-way function. (Hint: consider f_1 and f_2 that ignore half of the input bits given to them.)

- c. Suppose that $f_1 : \{0, 1\}^k \mapsto \{0, 1\}^k$ is a one-way function. Does it follow that $f(x) = f_1(1^k \oplus x)$ is a one-way function? If your answer is yes, prove it using a reduction. If your answer is no, give a contrived example for f_1 such that f is not a OWF.