Introduction to Modern Cryptography, Assignment #4*

Benny Chor and Rani Hod

Published: 29/12/2009; revised: 17/1/2010; due: 19/1/2010, in Rani's mailbox (Schreiber, 2nd floor).

This assignment contains four "dry" problems and one "wet" problem. Efficient solutions are always sought, but a solution that works inefficiently is better than none. The answers to the the "wet" problem should be given as the output of a `Sage`, `maple` or `WolframAlpha` session.

1. **Coin Flipping Over the Phone**

   In class, we will discuss the application of one way functions to enable two parties that do not trust each other to flip coins over the phone. We assume that the parties do not deviate from the *syntax* of the protocol. Still, they may try to cheat in different ways, provided the messages they send have the right format. For example, Alice could send a composite number where the protocol calls for a prime number.[1]

   In class, we will discuss a specific implementation using RSA, but more generally the protocol can be cast as follows: Let the two parties be good old Alice 'n Bob. Alice chooses $F : D \to D$ — a one-way, one-to-one function. Let $B : D \to \{0,1\}$ be an efficiently computable predicate on elements of D. Alice starts by sending a description of F and of B to Bob. The general protocol then proceeds as follows:

   1) Alice picks an $x \in D$, computes $y = F(x)$ and $b = B(x)$, and sends y to Bob (this is supposed to create a *commitment* to the value x).
   2) Bob sends to Alice his guess for $B(x)$, namely a bit $c \in \{0,1\}$. The bit c could either be the result of a coin flip or the outcome of some efficient algorithm applied by Bob in an attempt to guess $B(x)$.
   3) After receiving c, Alice sends x to Bob, who can now compute $b = B(x)$ on his own.
   4) If $c = b$ then Bob wins the coin toss and otherwise ($c \neq b$) Alice wins.

   The coin flip described in class had $D = \mathbb{Z}_N^*$, $F(x) = x^e \bmod N$, and $B(x)$ as the least significant bit of x.

   (a) Explain what goes wrong if the order of steps 1) and 2) is reversed.

   (b) Let $D = \mathbb{Z}_p^*$, $F(x) = g^x \bmod p$ where p is a prime number and g is a primitive element in $\mathbb{Z}_p^*$. Let $B(x)$ be the least significant bit of x. Show that Bob can win this game with probability 1.

   (c) Is the assumption that F is one-to-one necessary?

   (d) Give a *concrete example* where $F(x) = x^e \bmod N$ is not one-to-one and Alice can cheat and win the game with probability 1.

   (e) Give a *concrete example* where $F(x) = g^x \bmod p$ is not one-to-one and Alice can cheat and win the game with probability 1.

   Give an example where F is not one-to-one but neither Alice nor Bob can cheat. You can make reasonable assumptions if they do indeed make sense, but make sure to state them carefully.

2. **Hard Core Predicates**

   Let $F : D \to D$ be a one-way, one-to-one function. Let $B : D \to \{0,1\}$ be an efficiently computable predicate on elements of D. Let $\mathcal{A}$ be a blackbox (or "magic box") that on input $y = F(x)$ produces as output the bit

*Draft. Questions will not substantially change albeit might presented in a different wording.

[1] This specific attempt is not very smart since Bob will easily detect it.

$B(x)$. We say that B is a *hard core bit* for F if there is an efficient algorithm that inverts F, using $\mathcal{A}$ as a subroutine (each call to $\mathcal{A}$ is counted as one step). In class we stated (without proof) that $B(x) =$ "the least significant bit of x" is a hard core bit for the RSA function $F(x) = x^e \bmod N$ on $D = \mathbb{Z}_N^*$.

(a) Let $D = \mathbb{Z}_p^*$, $F(x) = g^x \bmod p$ where p is an odd prime number and g is a primitive element in $\mathbb{Z}_p^*$. Define

$$\mathrm{Half}_p(x) = \begin{cases} 0 & \text{if } 1 \le x < \frac{p-1}{2} \\ 1 & \text{if } \frac{p-1}{2} \le x \le p-1 \end{cases}$$

Show that $\mathrm{Half}_p(\cdot)$ is a hard core predicate for $F(x) = g^x \bmod p$.

(b) Let $D = \mathbb{Z}_N^*$, $F(x) = x^e \bmod N$ where $N = pq$, both p and q are prime numbers, and e is relatively prime to $\varphi(N)$. The numbers e and N are known, but N's factorization is not given. Define

$$\mathrm{Half}_N(x) = \begin{cases} 0 & \text{if } 1 \le x < N/2 \\ 1 & \text{if } N/2 \le x \le N-1 \end{cases}$$

Show that $\mathrm{Half}_N(\cdot)$ is a hard core predicate for $F(x) = x^e \bmod N$.

(c) Suppose $B : D \to \{0,1\}$ is a hard core predicate for F as defined above, and $F : D \to D$ is a one-way, one-to-one function. Is it true that the coin flipping protocol from Question 1 is indeed fair, i.e., no side has any non-negligible advantage compared to an unbiased coin? Provide a short proof if the answer is positive, and short, convincing evidence if your answer is negative.

3. **TMTO Success Probability Redux**

Let $f : X \to X$ be a random function; that is, $f(x)$ is selected independently at random from X for each $x \in X$. Our mission is to design a time-memory trade-off for inverting f: we precompute a data structure of size $O(m)$, with which, given $y \in X$, we can run an $O(t)$ algorithm trying to find $x \in X$ such that $f(x) = y$.

Our preprocessing step, as described in the recitation, consists of selecting m starting points $s_i \in X$ and calculating for each one the respective endpoint $e_i = f(f(\cdots f(s_i)))$ of the chain of f-applications. Formally, we define the ith chain $(x_i^j)_{j=0}^t$ as $x_i^0 = s_i$, $x_i^{j+1} = f(x_i^j)$, and $e_i = x_i^t$.

In the recitation we provided informal evidence that, for $m = t = |X|^{1/3}$, the data structure covers $\Omega(|X|^{2/3})$ elements, and hence the success probability for a single y is $\Omega(|X|^{-1/3})$. In this question we will elaborate on this lower bound.

- We say that the element x_i^j is *new* if it was not previously covered; that is, if $x_i^j \ne x_i^{j'}$ for $0 \le j < j'$ and $x_i^j \ne x_{i'}^{j'}$ for $1 \le i' < i$, $0 \le j' \le t$. Show that $Pr\left(x_i^j \text{ is new}\right) \ge e^{-ijt/|X|}$ and deduce that the expected number of covered elements is at least $\sum_{i=1}^m \sum_{j=1}^t e^{-ijt/|X|}$.
- The best we could hope for is to cover mt elements. Show that we actually cover $\Omega(mt)$ elements in expectance as long as we keep $mt^2 \le |X|$, so for $mt^2 = |X|$ the success probability is $\Omega(1/t)$.
- Argue that by increasing m or t further we gain no substantial increase.[2]

4. **Shamir's Secret Sharing**

Using `Sage` (or a different software of your choice), set up a system for 3-out-of-5 secret sharing scheme over the finite field $\mathbb{Z}_7$. Generate two different quadratic polynomials $f(x), g(x)$ that have different free terms $f(0) \ne g(0)$, yet $f(i) = g(i)$ for $i = 1, 2$.

In class, we argued that the secret can be expressed as a *linear combination* of the shares. Demonstrate this for two sets of participants: $\{1,2,3\}$ and $\{1,2,5\}$. For each set, compute explicitly the coefficients for extracting the secret. For example, in case of the first set, you should find the coefficients b_1, b_2, b_3 such that $h(0) = b_1 h(1) + b_2 h(2) + b_3 h(3)$ for every degree 2 polynomial. Find such coefficients c_1, c_2, c_5 for the second set of participants as well.

Demonstrate that for $f(x), g(x)$ chosen above, your linear combinations indeed work.

[2] Note that we will be arguing that *this lower bound*, rather than the actual success probability, does not improve for $mt^2 \gg |X|$. But the analysis is essentially tight.

5. **Reusing Randomness in El Gamal Signature Scheme**

   Recall that in El Gamal signature scheme, discussed in lecture 9, the signature generation is *randomized*. The signer is supposed to choose a new, independent random k for signing each message. Show that if the signer uses the same k for signing two different messages, then it is possible to extract the secret key, x, from two (signature, message) pairs and the public key. Write down equations for x given all this information. If you need some assumptions for solving the equations, state them and estimate how likely they are to be satisfied.