

Homework 2

*Instructor: Anna Lysyanskaya**Due: February 7, 2007***Problem 1: Electronic Voting**

In class we discussed a protocol for electronic voting when the ballot contains two choices: yes or no. We assume that we have a potentially biased third party (eg an election official), TP, who can be trusted not to reveal private votes, but who may attempt to skew the election results. The protocol goes as follows:

First we need a few tools:

A homomorphic encryption scheme is an encryption scheme which comes with a special operation \oplus such that $E_{\text{PK}}(x) \oplus E_{\text{PK}}(y) = E_{\text{PK}}(x + y)$, where E_{PK} is the encryption algorithm for public key PK.

Zero knowledge proofs allow a prover to prove that a statement is true without revealing any other information. Zero knowledge proofs exist for any problem in NP.

Voting protocol:

- TP chooses a public key PK and a secret key SK for a homomorphic public key encryption scheme. TP publishes PK.
 - Each voter encodes his vote as 1 if his vote is yes, and 0 if his vote is no. He encrypts the 0 or 1 using the public key PK, and posts $E_{\text{PK}}(0)$ or $E_{\text{PK}}(1)$. Finally, he gives a zero knowledge proof that the value he has encrypted is equal to 0 or 1.
 - Everyone uses the homomorphic operation to compute $E_{\text{PK}}(n)$, where n is the sum of all the 0,1 votes.
 - TP uses SK to decrypt $E_{\text{PK}}(n)$ and find n . He publishes n and gives a zero knowledge proof that this the correct decryption.
 - If n is more than half the total number of voters, the measure passes, otherwise it fails.
- a. How would you change this scheme if each voter was supposed to vote for one of three candidates?
 - b. In class we discussed a variation of this scheme where we have two election officials, TP1 and TP2. Each one publishes its own public key so we have PK1 and PK2. Now, to encode a yes, a voter will choose a random pair of numbers, a and b , such that $a + b = 10$ (choose a random a between 0 and 10, and then set $b = 10 - a$). To encode a no, a voter will choose a random a, b such that $a + b = 9$. The voter will then encrypt a under PK1 and b under PK2, and post them along with a proof that a and b sum to either 9 or 10. Everyone will compute the encryption of the sum of the a 's and the sum of the b 's. Finally, TP1 will reveal $n1$, the sum of all the a 's, and TP2 will reveal $n2$, the sum of all the b 's, and each will give a zero knowledge proof that this decryption is correct.
If N is the total number of voters, how would we compute the total number of yes votes?
 - c. Suppose we have a large number of voters who all follow this protocol. Assume that TP1 and TP2 do not communicate. Suppose that each voter votes yes with probability $1/2$ and no with probability $1/2$. Knowing this, can TP1 expect to correctly guess more than 50% of the votes? Describe his algorithm.

- d. Can we reduce the election official's probability of guessing Alice's vote by changing the range from which a and b are chosen?

Problem 2: Zero Knowledge Proof

In Lecture 1 we saw a protocol for running a physical proof for the 3-colorability problem:

Suppose a prover wants to convince a verifier that he knows how to 3-color a particular graph: that he can paint every vertex red, green, or blue so that no two vertices of the same color are adjacent. In our example, the prover will use physical props (paper, colored pens, paper cups). The actual GMW protocol does not use any physical props: instead of paper cups and colored pens, it uses appropriate cryptographic constructs.

The prover, in private, draws the graph on a piece of paper and colors all the vertices. The prover has 6 options for how to 3-color the graph: one option is the original 3-coloring that he knows to begin with, and the other five can be obtained by permuting the colors. He chooses a random one of the 6 options, and colors the graph accordingly. He then hides the vertices underneath the paper cups. Now the verifier can come in the room. The verifier chooses any two adjacent vertices and removes their paper cups. If the vertices are the same color, then the verifier knows the prover is lying. Otherwise, the verifier is satisfied.

If the prover knows a 3-coloring of the graph, then the verifier will always be satisfied. If the graph is not 3-colorable, then the verifier has a chance to catch the prover. The verifier and prover can repeat the protocol many times, so that if the prover is lying, the verifier is guaranteed to catch him with high probability. So this protocol is a convincing proof.

The reason that this protocol is a zero-knowledgeproof is that the verifier learns no information about the 3-coloring of the graph. Since the prover has 6 permutations of colors to choose from every time, the two colors the verifier sees are chosen uniformly at random from the set of red, green, blue. Once the verifier knows which vertices he wants to examine, he might as well have picked the colors himself.

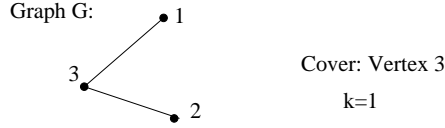
Now lets design a proof for a different NP-complete problem:

Definition 1 (Vertex Cover Problem). *Given a graph $G = (V, E)$ and an integer k , does there exist a size k vertex cover, i.e. a size k subset $C \subset V$ such that for all edges $e \in E$ at least one endpoint is in C ?*

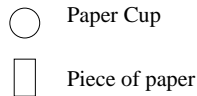
Suppose Alice and Bob share a graph G . Alice claims that G has a size k vertex cover. Alice will attempt to prove it as follows (see figure below for example):

- Alice sends Bob out of the room.
- On one side of the table, Alice lists the graph's vertices in random order. We assume that each vertex of the original graph is labeled with unique number from 1 to $|V|$. Alice now labels each vertex with this number in green and covers the label with a paper cup. She then gives each vertex a new random label (written in blue and not covered by paper cups).
- On the other side of the table, Alice draws a bunch of parallel lines, one for each edge. She labels the endpoints using the new (blue) vertex labelling. Finally, she covers these labels with paper cups.
- On the vertex list, next to each vertex that is in the vertex cover set C , Alice writes a "C". Next to all other vertices, Alice writes a "Not C". She then covers all the "C"'s and "Not C"'s with paper cups.
- On the edge list, next to each edge, Alice draws an arrow that points to an endpoint which is in the cover. She covers this arrow with a piece of paper.

- Alice calls Bob back into the room.



Alice will draw the following on the desk:



How can Bob verify this proof? We're going to divide the solution into several small steps.

- Suppose Bob only wanted to check that the graph is represented correctly (i.e. that the graph is the same one, G , that he and Alice have agreed on.) How can he confirm this without learning any other information?
Hint: Recall that the graph isomorphism problem (given graphs G_1, G_2 , determine whether there is a relabelling of the vertices of G_1 which make it identical to G_2) is considered hard.
- Now suppose Bob only wants to check that the cover has the appropriate size (the agreed upon k). How can he confirm this without learning any other information?
- Finally, suppose Bob only wants to check that each edge is covered. How can Bob do this without learning any other information (he is allowed to examine only one edge in each round)? With what probability is he guaranteed to catch Alice if she does not know a k -cover?
- What should Bob's overall strategy for verifying Alice's vertex cover proof be, and what is the minimum probability that Alice will be caught if she cheats? (Hint, the strategy will probably have to be randomized.)
- Assuming Alice is willing to repeat this process as many times as necessary, how many times should Bob run this algorithm so that if Alice cheats she will be caught with probability at least $O(1)$?

Problem 3: Shannon security

Recall the Shannon definition of security for a cryptosystem.

Let G be an algorithm that generates the secret key s and gives it to Alice and Bob. Let E and D be the encryption and decryption algorithms. Let us think of the plaintext message m as drawn from some probability distribution M . Let $c = E(m, s)$ be a ciphertext.

Definition 2 (Shannon-security). (G, E, D) constitute a Shannon-secure cryptosystem if for all messages $m \in M$, and for all possible ciphertexts c , $\Pr[m] = \Pr[m|c]$.

In other words, Shannon-security says that all that the adversary knows to begin with, is that the message came from the message space M . Seeing the ciphertext did not make him any the wiser.

a. Consider the following definition:

Definition 3 (A-security). (G, E, D) constitute an A-secure cryptosystem if for all messages $m \in M$, and for all possible ciphertexts c , $\Pr[c] = \Pr[c|m]$.

Is A-security equivalent to the original Shannon definition? Prove or disprove your answer.

b. Suppose we have a Shannon-secure cryptosystem, such that for some distribution M of messages, \forall messages m_0, m_1, \forall possible ciphertexts c , $\Pr[m_0|c] = \Pr[m_1|c]$. What (if anything) can you infer about distribution M ? Prove your answer.

Problem 4: Negligible functions

In cryptography, we usually define security by requiring that the probability of some undesirable event (e.g. Eve guesses the message) to be so small that one would never notice it. To that end, we define a negligible function as follows:

Definition 4. (Negligible function) A function $\nu(k) : \mathbb{N} \mapsto [0, 1]$ is called negligible if for all polynomials p , there exists some $k_0 \geq 1$ such that for all $k > k_0$, $\nu(k) < |1/p(k)|$.

In this problem we will develop some intuition for this useful concept and how to work with it.

- a. Give an example of a negligible function $\nu(k)$ where $\nu(k) > 0$ for all k .
- b. Suppose that ν , is a negligible function. Let p be a polynomial such that $p(k) \geq 0$ for all $k > 0$. Which of the following functions are negligible: (give yes/no)
- $\nu(p(k))$.
 - $\nu_1(k) * \nu_2(k)$ where both ν_1 and ν_2 are negligible.
 - $\nu(k) * p(k)$.
 - $\sum_{i=1}^{p(k)} \nu_i(k)$ where each ν_i is a negligible function.
 - $\nu_1(k)/\nu_2(k)$ where both ν_1 and ν_2 are negligible.
 - $\frac{1}{p(k)} - \nu(k)$.
- c. Suppose that $\epsilon : \mathbb{N} \mapsto [0, 1]$ is not a negligible function. Does it follow that for some polynomial p (where $p(k) > 0$ for all k) and some k_0 , $\epsilon(k) > 1/p(k)$ for all $k > k_0$? If your answer is yes, prove it. If your answer is no, give a counter-example.