Problem 1: Zero Knowledge Proof

In class we saw a physical and a cryptographic protocol for a zero-knowledge proof for the 3-colorability problem. Now let’s 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? That is, does there exists 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 in the following manner (see Figure 1 for a visualization).

- Alice sends Bob out of the room.
- Assume that each vertex of the original graph is labelled with unique number from 1 to \( |V| \). Alice lists these labels in a random order in green, and covers them each with a paper cup. Next to these, she writes a new random label in blue, 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.

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

a. 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 is considered hard. In the graph isomorphism problem, you are given graphs \( G_1, G_2 \), and you must determine whether there is a relabelling of the vertices of \( G_1 \) which make it identical to \( G_2 \).

b. 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?

c. 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?
Alice will draw the following on the desk:

![Graph G](image)

**Figure 1:** An example of the protocol setup.

d. 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.

e. 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 2: A Not Zero Knowledge Proof**

Consider the following protocol.

**Input:** RSA modulus \( n \), and a value \( x \in \text{QNR}_n \).

The prover knows the factorization of \( n \) and wants to prove that \( x \) is in \( \text{QNR}_n \).

1. The verifier (\( V \)) forms a challenge value as follows: choose \( c \leftarrow \{0, 1\} \), choose random \( r \in Z_n^* \). 
   \( V \) sends \( y = r^2 x^c \) to \( P \).
2. The prover uses the factorization of $n$ to determine whether $y \in \text{QR}_n$. If so, let $c' = 0$. Otherwise, let $c' = 1$. Send $c'$ to $V$.

3. $V$ checks that $c' = c$. If so, $V$ accepts, otherwise $V$ rejects.

It turns out that this protocol is sound, but it is NOT zero-knowledge.

a. Prove that this protocol satisfies the soundness property.

b. Explain where the proof of ZK property would break down. Why wouldn’t we be able to build a valid simulator? Note the simulator would not know $p, q$, as these are not known to the verifier.

c. Explain how a cheating verifier could use an honest prover to learn something.

**Problem 3: Bit Commitment from PRGs**

A bit-commitment scheme is a protocol between two parties: the committer $C$ and the receiver $R$, where $C$ commits to a single bit value $b \in \{0, 1\}$. The protocol consists of two phases:

- During the **commitment phase** $C$ and $R$ interact in such a way that $C$ has committed to his secret choice $b$.
- In the **opening phase**, $C$ reveals his secret choice $b$.

In order for the protocol to be secure, informally the following need to hold: (a) after the commitment phase, the receiver shouldn’t be able to learn anything about the committer’s secret bit $b$ (hiding), (b) the committer shouldn’t be able to change his mind after he has send his commitment to $R$ (binding).

Consider the following construction. Let $G$ be a PRG: $\{0, 1\}^k \rightarrow \{0, 1\}^{3k}$.

**Commitment Phase** $R$ sends $z \leftarrow \{0, 1\}^{3k}$ to the committer. Then, $C$ generates $s \leftarrow \{0, 1\}^k$ and computes $c$ to be $c = G(s)$ if $b = 0$ and $c = G(s) \oplus z$ if $b = 1$. $C$ sends $c$ to the receiver $R$.

**Opening Phase** $C$ sends $s$ and $b$ to the receiver $R$. If $b = 0$, $R$ checks that $c = G(s)$; else $R$ checks that $c = G(s) \oplus z$. If the check succeeds, then $R$ outputs $b$, otherwise it outputs $\perp$ (to indicate failure).

a. Prove that the scheme is hiding, i.e., the receiver $R$ cannot output a correct guess for $b$ with non-negligible advantage.

**Hint:** Construct a reduction that contradicts the pseudorandomness of $G$.

b. Prove that the scheme is binding, i.e. for all ppt $A$, there exists a negligible $\nu(k)$ such that

$$\Pr [z \leftarrow \{0, 1\}^{3k} ; (s_0, s_1) \leftarrow A(z) : c = G(s_0) = G(s_1) \oplus z] \leq \nu(k).$$

**Hint:** How many $z$’s are there that allow even an unbounded committer to change his mind? What is the probability of picking such a bad $z$?
Problem 4: Oblivious Transfer

Suppose that Alice has two binary messages $m_0$ and $m_1$, and that she wishes to transfer one of them to Bob obliviously, i.e., without learning which message Bob received and without Bob learning more than one of her messages. Bob has an input bit $b$ indicating that he wishes to learn $m_b$. Here is a candidate construction to achieve this goal. This construction assumes that $m_0$ and $m_1$ are bits.

1. Bob generates a Blum integer $n$, and retains its factorization $n = p \cdot q$. If $b = 0$, Bob sends Alice a value $y \in \text{QNR}_n$. If $b = 1$, Bob sends Alice $y \in \text{QR}_n$. He also sends $n$ to Alice. Alice verifies that $y$ has Jacobi symbol 1 (so $y$ is either QR or QNR).
2. Bob proves to Alice, using a ZK proof, that $n$ is a product of at most two distinct primes. (We will discuss below how this can be done.)
3. Alice and Bob engage in a zero-knowledge proof (outlined below) that the value $x = (-y) \cdot y$ is in $\text{QNR}_n$.
4. If Alice is satisfied with the proofs, she picks random values $r_1, r_2 \leftarrow \mathbb{Z}^*_n$ and sends $y^{m_0} \cdot r_2$ and $(-y)^{m_1} \cdot r_1^2$ back to Bob.

Our goal is to show that this interaction is an oblivious transfer. Recall that to prove this we need to satisfy the following three properties:

**Correctness** At the end of the transfer, Bob knows the message $m_b$ for his chosen bit $b$.

**Alice’s Security** Bob does not learn anything about the message $m_{\bar{b}}$, where $\bar{b} = b \oplus 1$. To that end, show that Bob obtains identical views in the case of $m_\bar{b} = 0$ and $m_\bar{b} = 1$.

**Bob’s Security** Alice does not learn Bob’s bit $b$; i.e., does not learn which of her messages Bob actually received. To that end, show that the view Alice receives in this interaction when $b = 0$ is indistinguishable from the view she receives when $b = 1$.

Before we can prove that these three properties hold, we need to define our zero-knowledge protocol.

a. First, let’s try to use the protocol in Problem 2 above, which we showed was not zero-knowledge. If this protocol is used in Step 3, how is Bob’s security broken? In particular, describe an attack that Alice can use to learn Bob’s bit $b$.

b. Just because the protocol from Problem 2 isn’t zero-knowledge doesn’t mean we can’t make it zero-knowledge. Recall that the protocol works over the group $\mathbb{Z}_n^*$, where the prover knows the factorization $n = p \cdot q$ of the modulus, the verifier does not, and the prover is trying to convince the verifier that a certain value $x$ is in $\text{QNR}_n$. Using a perfectly binding, computationally hiding commitment scheme over values in $\mathbb{Z}_n^*$, we can fix the protocol by adding two extra steps as follows:

   (a) The verifier picks $c \leftarrow \{0, 1\}$ and $r \leftarrow \mathbb{Z}_n^*$ and sends $R = r^2 x^c$ to the prover.

   (b) The prover uses $p$ and $q$ to determine whether or not $R \in \text{QR}_n$. If so, let $c' = 0$. If not, let $c' = 1$. Now, instead of sending $c'$ in the clear, the prover sends a commitment $C = \text{Commit}(c')$.

   (c) Upon receiving $C$, the verifier sends back the values $r$ and $c$.

   (d) The prover checks that $R = r^2 x^c$. If this check passes, he sends back the opening to the commitment. If not, the protocol terminates.
(e) The verifier checks that the opening is valid for the commitment $C$, and also that $c' = c$. If both these checks pass, the verifier accepts. Otherwise, he rejects.

The completeness and soundness properties follow directly from the solution to Problem 2 and the fact that $\text{Commit}$ is perfectly binding, so they are still satisfied. We now just need to show that this improved protocol does in fact satisfy the zero-knowledge property. To do this, construct a simulator $S$ that, on input the modulus $n$ and the value $x$, interacts with the verifier in a way that is indistinguishable from the actual interaction with the prover.

c. Now, using the improved zero-knowledge proof in part b. (repeated $k$ times to reduce soundness error to $2^{-k}$), prove that the resulting OT protocol is secure; i.e. that it satisfies the three properties outlined above.

d. Explain how to modify the construction to handle bit strings $m_0$ and $m_1$, rather than bits.