Please list your collaborators and provide any references that you may have used in your solutions. Submit your homework here by Tuesday 24 November.

Question 1

In this question you will analyze the following bit commitment protocol based on a pseudorandom generator $G\colon \{0,1\}^k \to \{0,1\}^{3k}$. First, receiver picks a random string $R \in \{0,1\}^{3k}$ and shares it with sender. To commit to a bit s, sender chooses a random X and sends $G(X) + s \cdot R$ (i.e., $G(X)$ when $s = 0$ and $G(X) + R$ when $s = 1$). To reveal, sender reveals s and X and receiver checks that his commitment C equals $G(X) + s \cdot R$.

(a) Prove that if G is a pseudorandom generator then the commitment is hiding. Work out the parameters.

(b) Show that with probability $1 - 2^{-k}$ over the choice of R there does not exist a pair of inputs X and X' such that $G(X) + G(X') = R$. (**Hint:** Take a union bound over all pairs.)

(c) Prove that the commitment is binding. Work out the parameters.

Question 2

Bob has some database D that Alice wants to query, but she suspects that Bob might not give her correct answers. To ensure integrity Alice also has a short collision-resistant hash $h(D)$ of the database. When Alice wants to retrieve the contents $D(x)$ of database row x, Bob sends Alice the whole database D and she can verify that the hash is correct. This is impractical when the database is large. In this problem you will model this scenario cryptographically and explore a more efficient solution based on Merkle trees.

A database is a function $D\colon \{1, \dots, R\} \to \{0,1\}^n$ that maps a row x to a data item $D(x)$. A *succinct commitment protocol* has the following format. Alice has no input and Bob's input is the database D. In the setup phase, Bob sends Alice a commitment *com* to the database. In the query phase,

1. Alice sends a query $x \in \{1, \dots, R\}$ of her choice to Bob.
2. Bob returns an answer $y = D(x)$ and a certificate *cert*.
3. Upon receiving y and *cert*, Alice runs a verification which accepts or rejects.

The functionality requirement is that when Bob is honest Alice accepts with probability 1.

(a) Give a definition of (s, ε)-security. The adversary is a cheating Bob.[1] You may assume the availability of a random public key K available to all the parties (as in the collision-resistant hash setup).

(b) Let $com = h_K(D)$ and $cert = D$ where h is a collision-resistant hash function. Describe the verification and prove that the protocol is secure.

[1] There is no need for a "learning phase" as there is no secret information to be learned.

(c) The certificate in part (b) is nR-bits long. Now assume h is the Merkle tree-based collision resistant hash of depth $\log R$ from Lecture 6. Describe a different certificate of length $n(\log R + 1)$, the corresponding verification, and prove that the protocol is secure.
(**Hint:** It is sufficient for Bob to reveal the hashes at $\log R + 1$ nodes in the Merkle tree.)