Question 1

Consider the following encryption algorithm based on the shortLWE assumption. The secret key is a shortLWE secret $x \sim \nu^n$ and the public key is $PK = (A, Ax + e)$, where A is a random $n \times n$ matrix over $\mathbb{Z}_q$ and $e \sim \nu^n$. The encryption of a message represented by $M \in \mathbb{Z}_q$ under public key $PK = (A, b)$ is

$$Enc(PK, M) = (e' + x'A, e'' + x'b + M), \qquad x' \sim \nu^n, e' \sim \nu^n, e'' \sim \nu.$$

(A is a matrix, x, e, b are column vectors, x', e' are row vectors, and e'', M are scalars.)

(a) Give the corresponding decryption algorithm. Show that the scheme is functional assuming that the message is encoded in the $\log q - \log n - 2\log b - O(1)$ most significant bits of M.

Solution: Given a ciphertext (y', C) the decryption algorithm calculates $C - y'x$ and rounds it to the closest message in $\mathbb{Z}_q$. When $y' = e' + x'A$ and $C = e'' + x'(Ax + e) + M$, $C - y'x$ equals $M + e'' + x'e - e'x$. The entries of x, x', e, e', and e'' are bounded by b so $e'' + x'e - e'x$ has magnitude at most $2nb^2 + b$. Under the assumed encoding M is the unique possible message that differs from C by a number of this magnitude.

(b) Prove that the scheme is (s', ε')-message simulatable under the (s, ε)-shortLWE assumption. (Calculate the dependence of s' and ε' on s, ε, and other relevant parameters.)

Solution: The simulator outputs random independent values in $\mathbb{Z}_q$. Assuming (s, ε)-shortLWE The random variable

$$(PK, Enc(PK, M)) = (A, Ax + e, e' + x'A, e'' + x'(Ax + e) + M)$$

is $(s - t, \varepsilon)$-indistinguishable from

$$(A, r, e' + x'A, e'' + x'r + M),$$

where t is the size of the distinguisher that on input (A, y) samples x', e', e'' and outputs $(A, y, e' + x'A, e'' + x'y + M)$. Disregarding M this is a shortLWE instance with matrix $[A|r]$ and secret x', so it is (s, ε)-indistinguishable from $(A, r, r', r'' + M)$ for random r', r''. The distribution of the last random variable is independent of M and identical to the simulator's output.

By the triangle inequality, the encryption scheme is $(s - t, 2\varepsilon)$-message simulatable where t is the size of D, namely the complexity of $O(n^2)$ additions over $\mathbb{Z}_q$ once x', e', and e'' are fixed to maximize the distinguisher's advantage.

Question 2

In this question you will analyze the following LWE-based public-key identification protocol. The secret key is a random $x \sim \{-1, 1\}^m$. The public key is (A, xA) where A is a random $m \times n$ matrix over $\mathbb{Z}_q$. All arithmetic is modulo q.

1. Prover chooses a random $r \sim \{-b, \ldots, b\}^m$ and sends rA.
2. Verifier sends a random bit $c \sim \{0, 1\}$.
3. Prover sends $r + cx$.

(a) Show that if $m = 1$ then r conditioned on $|r| \leq b - 1$ and $r + x$ conditioned on $|r + x| \leq b - 1$ are identically distributed.

Solution: Both random variables are uniform over $\{-b+1, \ldots, b-1\}$. Since r is uniform, any pair of outcomes is equally probable under the conditioning. For $r + x$, any outcome y can arise from $r = y + 1$, $x = -1$ or from $r = y - 1, x = 1$ so any two outcomes are also equiprobable.

(b) Now let m be arbitrary as in the protocol. Show that r and $r + x$ are $O(m/b)$-statistically close.

Solution: For both random variables in part (a), the probability that the condition is not satisfied equals $2/(2b + 1)$. Since the entries of r and $r + x$ are independent, the probabilities that some coordinate falls outside the range $\{-b + 1, \ldots, b + 1\}$ are equal for r and $r + x$ and they are both upper bounded by $2m/(2b + 1) \leq m/b$. Conditioned on this not happening the two are identically distributed, so the advantage of any distinguisher is at most m/b.

(c) Show that the view of an eavesdropper who sees q' protocol transcripts is $O(q'm/b)$-statistically close to some random variable that can be efficiently sampled by a simulator that is given only the public key.

Solution: Let's do $q' = 1$ first. The view of the eavesdropper consists of the public key $(A, h = xA)$ and the transcript $(rA, c, r + cx)$. Given the public key the simulator samples y from $\{-b, \ldots, b\}^m$, $c \sim \{0, 1\}$, and outputs $(yA - ch, c, y)$. When $c = 0$ the two views are identically distributed. When $c = 1$ the two views are $(A, xA, rA, 1, r+x)$ and $(A, xA, (y-x)A, 1, y)$. If these two random variables were distinguishable, so would be $(x, r + x)$ and (x, y). In part (b) we showed these two are m/b-statistically close. Since c is equally likely to take values zero and one, the eavesdropper's view and the simulated view are $m/2b$-statistically close.

For general q' the simulator samples each transcipt independently given the public key. By the same argument, if the two views can be distinguished with advantage ε so can $(x, r_1+c_1x, r_2+c_2x, \ldots, r_{q'}+c_{q'}x)$ and $(x, y_1, \ldots, y_{q'})$. By a hybrid argument $(x, r_i + c_ix)$ and (x, y_i) can then be distinguished with advantage ε/q' for some i. It follows that $\varepsilon \leq q'm/2b$.

(d) Let $h_A(x) = xA$, where the entries of x are of magnitude at most $2(b+1)$. Show that if h is a collision-resistant hash function then no efficient cheating prover can handle both challenges $c = 0$ and $c = 1$. Conclude that, if repeated sufficiently many times, the protocol is secure against eavesdropping. (Work out the dependences between the security parameters.)

Solution: This question doesn't make sense without specifying when the honest verifier accepts. The verifier checks that the public key and the prover's messages satisfy the desired linear relation: If the first message is $k = rA$ and the third one is $y = r + cx$ then yA should equal $k + ch$. The verifier checks this, but it also checks that the entries of y are in the range $\{-b-1, \ldots, b+1\}$. Both conditions are satisfied by the honest prover.

Now suppose a cheating prover can handle both $c = 0$ and $c = 1$ with responses y_0 and y_1, respectively. Then $y_0, y_1 \in \{-b-1, \ldots, b+1\}^n$ both satisfy the equation $y_c A = k + ch = k + cxA$. Subtracting these two equations gives $(y_1 - y_0)A = xA$.

We can almost conclude that the cheating prover found a collision to h_A, except that x and $y_1 - y_0$ might be equal. It turns out that this can happen at most a $q^n/2^m$-fraction of the time. The reason is that for a typical choice of x, h_A will have collisions $x' \neq x$ such that $h_A(x') = h_A(x)$. Conditioned on the public key, each of these colliding values is equally likely to be the secret key, so the cheating prover could not have found x unless it guessed it correctly among all of them. If h_A was regular (i.e. each output arises from the same number of inputs) then the probability that the cheating prover "guessed" the acutal secret key x is at most $q^n/2^m$. (If not, the probability can only be lower, but I won't prove it.)

In conclusion, if the cheating prover succeeds with probability at least $(1+\varepsilon)/2$, then it can handle both $c = 0$ and $c = 1$ with probability at least ε and it can find a true collision to h_A with probability at least $(1 - q^n/2^m)\varepsilon$. If say $\varepsilon = 1/2$ and $m = n \log q + 1$ and the prover is run k times, then any prover that passes validation with probability $(3/4)^k$ can be used to find a collision in h_A with probability $1/4$.

(e) **(Optional)** Prove that the protocol is secure against impersonation.

Solution: (Sketch) The simulator (for each round of impersonation) acts like the simulator in part (b), then runs the cheating verifier and outputs the transcript if his actual challenge c^* equals the guessed challenge c. The message received by the cheating verifier in round 1 is no longer independent of c because the distributions of the first messages are a bit different when $c = 0$ and when $c = 1$. Nonetheless it can be argued that the event $c^* = c$ still occurs with probability at least $1/2 - O(m/b)$. In conclusion, after r simulation attempts the simulator produces a transcript that is $O(m/b)$-indistinguishable from the real one with probability at least $(1/2 - O(m/b))^r$.

Question 3

In this question you will show that using an obfuscator, an adversary can plant a collision in a hash function that makes it insecure against him, but secure against everyone else. Let $h\colon \{0,1\}^m \to \{0,1\}^n$ be a collision-resistant hash, Obf an obfuscator, and A the following algorithm:

1. Sample a random key K and a random input $\hat{x} \sim \{0,1\}^m \setminus \{0\}$.
2. Construct a circuit h' that implements the function

$$h'(x) = \begin{cases} h_K(0), & \text{if } x = \hat{x}, \\ h_K(x), & \text{if not.} \end{cases}$$

3. Output $H = Obf(h')$.

Then A knows a collision for H, namely the pair $(0, \hat{x})$. We can view H both as a random key and the function described by it, so (s, ε)-collision-resistance means that the probability that $C(H)$ outputs a collision for H is at most ε for every C of size at most s.

(a) Show that the views D^{h_K} and $D^{h'}$ are $q/(2^m - 1)$-statistically close for any distinguisher D that makes at most q queries to its oracle.

Solution: Even for fixed K, h_K and h' differ on a single input random input $\hat{x}$. Consider a hybrid h_i that answers its first $i-1$ queries like h_K and the rest like h'. Then D^{h_i} and $D^{h_{i+1}}$ are $1/(2^m - 1)$-statistically close because they can only be distinguished if D queries $\hat{x}$ in round i, which happens with probability $1/(2^m - 1)$. By the triangle inequality the views D^{h_K} and $D^{h'}$ are $q/(2^m - 1)$-statistically close.

(b) Show that if h is (s, ε)-collision resistant and Obf is $(s + 2t + O(m), \varepsilon')$-VBB secure, H is $(s - tt', \varepsilon + \varepsilon' + q/(2^m - 1))$-collision resistant, where t and t' are the sizes h and the VBB simulator, respectively.

Solution: Suppose some collision-finder C on input H, finds a collision x, x' for H with probability ε^*. By the VBB security of H there is a simulator D of size s such that the outputs of $S^{h'}$ and $C(H)$ are $(2t + O(m), \varepsilon')$-indistinguishable. This means the probabilities that $C(H)$ and $D^{h'}$ output collisions can differ by at most ε', because a distinguisher that tests if x, x' is a collision has size $2t + O(m)$. In conclusion, the probability that $D^{h'}$ outputs a collision for h' is at least $\varepsilon^* - \varepsilon'$.

By part (a) the probability that D^{h_K} outputs a collision for h' is then at least $\varepsilon^* - \varepsilon' - q/(2^m - 1)$. Since h_K is independent of $\hat{x}$, this collision involves $\hat{x}$ with probability at most $1/(2^m - 1)$. Otherwise, D^{h_K} must have output a collision for h_K. This happens with probability $\varepsilon^* - \varepsilon' - (q+1)/(2^m - 1)$ (a little worse than advertised).

It remains to do a bit of technical work: An actual collision-finder takes the key K as an input. We want to use it to implement D^{h_K}, namely provide oracle access to h_K. As there are at most t' oracle calls, each implementable in size t, this results in a circuit that is larger by tt', accounting for the deterioration in the size parameter.

(c) Show that the MAC from Theorem 5 in Lecture 6 is insecure against a forger that knows $\hat{x}$.

Solution: A forger that knows $\hat{x}$ can query the oracle on $\hat{x}$ and obtain the tag $T = Tag(H(\hat{x}))$. Then the message-tag pair $(0, T)$ is a forgery because $H(\hat{x}) = H(0)$.

Question 4

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 $D(r)$ of database row r, 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 $D\colon \{1, \ldots, R\} \to \{0,1\}^n$ that maps a row x to a data item $D(x)$. A *database 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 $x \in \{1, \ldots, R\}$ 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).

Solution: (s, ε)-security means that for every cheating Bob of size at most s, for any given x, the probability that $y \neq D(x)$ and Alice accepts is at most ε.

(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.

Solution: Verification checks that $h_K(cert) = com$ and $y = cert(x)$. If the scheme is not (s, ε)-secure and if Alice accepts some $y \neq D(x)$, then *cert* and D cannot be equal (because they differ at y) but they have the same hash, so cheating Bob has found a collision for h_K, so h_K is not (s, ε)-collision resistant.

(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.)

Solution: The certificate is specified recursively. For a tree of depth 0 the certificate is $y = D(x)$ itself. For a tree of depth $i + 1$, if y is in the left subtree the certificate consists of the recursive certificate for this subtree together with the Merkle tree hash of the right half of the database. If y is in the right subtree it is the opposite.

Verification is also performed recursively. For depth zero *cert* is checked against the commitment $com = y$. For larger depth, a candidate hash for the subtree involving y is computed recursively. Together with the claimed value for the hash of the other half, these are plugged into the root hash and the verifier accepts if the value equals the commitment.

We argue security. When the depth is zero the scheme is perfectly secure. Now suppose a cheating Bob makes Alice accept for some $y \neq D(x)$. One possibility is that this cheating Bob already succeeds when his input is the half of the database containing row x. If not, then it must be that claimed value for the hash of this subtree is different from the true value, so if Alice accepts Bob must have found a collision for the root hash.

Quantitatively, if the depth-d scheme is not (s', ε')-secure, then either the depth-$(d-1)$ scheme is not $(s', \varepsilon' - \varepsilon)$-secure or the root hash is not $(s' + v, \varepsilon)$-collision resistant where v is the size of Alice's verification circuit (for depth d). Unwinding this recursion, we get that if the base hash function is (s, ε)-collision resistant then the scheme is $(s - v, d\varepsilon)$-secure.

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