CS65500: Advanced Cryptography Instructor: Aarushi Goel

Homework 6

Due: April 25; 2025 (11:59 PM)

1 Private Set Intersection

(20 Points) Recall the following PSI protocol discussed in class and **formally prove that it satisfies semi-honest security with respect to both parties, Alice and Bob.**

Setup. Alice has input $X = \{x_1, \ldots, x_m\}$. Bob has input $Y = \{y_1, \ldots, y_n\}$. Both parties agree on a group $(\mathbb{G}, p, g)$. Both parties agree on a hash function $H : \{0,1\}^* \to \mathbb{G}$ modeled as a random oracle, which maps strings of arbitrary length into *random* group elements.

Alice $\to$ Bob.

1. Alice picks a random number $\alpha \xleftarrow{\$} \mathbb{Z}_p$;
2. For each $i \in [m]$, Alice computes $a_i = H(x_i)^\alpha$;
3. Alice sends $A = \{a_1, \ldots, a_m\}$ to Bob;

Bob $\to$ Alice.

1. Bob picks a random number $\beta \xleftarrow{\$} \mathbb{Z}_p$;
2. For each $j \in [n]$, Bob computes $b_j = H(y_j)^\beta$;
3. For each $i \in [m]$, Bob computes $c_i = a_i^\beta$.
4. Bob sends $B = \{b_1, \ldots, b_n\}$ and $C = \{c_1, \ldots, c_m\}$ to Alice;

Output Computation.

1. Initialize a set $Z = \emptyset$;
2. For each $j \in [n]$, Alice computes $d_j = b_j^\alpha$;
3. For each $i \in [m]$ such that there is a $j \in [n]$ such that $c_i = d_j$, Alice adds x_i to Z;
4. Alice computes and outputs Z.

2 Pseudorandom Correlation Generator

(15 Points) In class, we learned how to design a two-party pseudorandom correlation generator (PCG) for the VOLE correlation, where one party upon expansion of their seed obtains pseudorandom vectors $\vec{u}$ and $\vec{v}$, and the other party upon expansion of their seed obtains a random scalar x and a pseudorandom vector $\vec{w}$, such that $\vec{u} \cdot x + \vec{v} = \vec{w}$. Describe how this construction can be adapted to generate a PCG for the following Beaver triple-style correlation: $(\vec{a}, b, \vec{c})$, where b is a random scalar and $\vec{a}, \vec{c}$ are pseudorandom vectors satisfying $\vec{a} \cdot b = \vec{c}$. Each party upon expansion of their seeds should obtain an additive secret sharing of $\vec{a}$, b, and $\vec{c}$. Argue correctness of your construction. You **do not** need to prove its security.

(**Hint:** Observe that in a VOLE correlation, the pair $-\vec{v}, \vec{w}$ can be interpreted as additive shares of $\vec{u} \cdot x$. Therefore, by letting $\vec{a} = \vec{u}$ and $b = x$, the PCG discussed in class already enables the parties to obtain additive shares of $\vec{c}$. The only remaining task is to modify this construction so that it also yields additive secret shares of b and $\vec{a}$ – rather than having one party learn $\vec{a}$ and the other learn b in the clear.)

3 Homomorphic Encryption

(20 points) Consider a special linearly homomorphic secret-key encryption scheme that satisfies the following definition.

Definition 1 *For all $\lambda \in \mathbb{N}$, a CPA-secure special additively homomorphic secret-key encryption comprises of a tuple of PPT algorithms* $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Refresh}, \mathsf{Linfunc})$ *defined as follows:*

- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*: The key generation algorithm takes the security parameter 1^λ as input and outputs a public-key* pk *and a secret-key* sk.
- $ct \leftarrow \mathsf{Enc}(\mathsf{sk}, m; r)$*: The encryption algorithm takes as input, the secret-key* sk*, a message $m \in \{0,1\}^n$ and a random string $r \in \{0,1\}^\lambda$, and outputs a ciphertext ct.*
- $m \leftarrow \mathsf{Dec}(\mathsf{sk}, ct)$*: The decryption algorithm takes as input the secret-key* sk *and a ciphertext ct, and outputs a message m.*
- $ct' \leftarrow \mathsf{Refresh}(\mathsf{pk}, ct; r)$*: The refresh algorithm takes as input the public key* pk*, a ciphertext ct and a random string $r \in \{0,1\}^\lambda$, and outputs a new ciphertext ct'.*
- $ct' \leftarrow \mathsf{LinFunc}(\mathsf{pk}, f, ct_1, \ldots, ct_k)$ *: This algorithm takes as input the public key* pk*, a list of $k \geq 1$ ciphertexts $ct_1, \ldots, ct_k$ and a linear function $f : (\{0,1\}^n)^k \to \{0,1\}^n$ and outputs a new ciphertext ct'.*

These algorithms satisfy the following:

1. ***Correctness:*** *Let $(\cdot, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, then $\forall m \in \{0,1\}^n$ and uniformly sampled $r \leftarrow_\$ \{0,1\}^\lambda$, it holds that:*

$$\Pr\left[m \leftarrow \mathsf{Dec}\left(\mathsf{sk}, \mathsf{Enc}(\mathsf{sk}, m; r)\right)\right] = 1$$

2. ***IND-CPA Security:*** *Let* $(\cdot, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, then* $\forall \{m_{0,i}, m_{1,i}\}_{i \in \mathsf{poly}(\lambda)} \in \{0,1\}^n$*, the following two distributions are computationally indistinguishable:*

$$\left\{\{\mathsf{Enc}(\mathsf{sk}, m_{0,i}; r_i)\}_{i\in\mathsf{poly}(\lambda)}; \{r_i\}_{i\in\mathsf{poly}(\lambda)} \leftarrow\$ \{0,1\}^\lambda\right\}$$

$$\left\{\{\mathsf{Enc}(\mathsf{sk}, m_{1,i}; r_i)\}_{i\in\mathsf{poly}(\lambda)}; \{r_i\}_{i\in\mathsf{poly}(\lambda)} \leftarrow\$ \{0,1\}^\lambda\right\}$$

3. ***Re-randomization:*** *Let* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, then* $\forall m \in \{0,1\}^n$*, and any uniformly sampled* $r \leftarrow\$ \{0,1\}^\lambda$*, the following two distributions are computationally indistinguishable:*

$$\left\{\mathsf{Refresh}(\mathsf{pk}, ct; r'); r \leftarrow\$ \{0,1\}^\lambda\right\}$$

$$\left\{\mathsf{Enc}(\mathsf{sk}, \mathsf{Dec}(\mathsf{sk}, ct); r'); r \leftarrow\$ \{0,1\}^\lambda\right\},$$

where $ct \leftarrow \mathsf{Enc}(\mathsf{sk}, m; r)$.

4. ***Linear Homomorphism:*** *Let* $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$*, then for any* $k \geq 1$*,* $\forall \{m_i\}_{i\in[k]} \in \{0,1\}^n$*, any uniformly sampled* $\{r_i\}_{i\in[k]} \leftarrow\$ \{0,1\}^\lambda$*, and any linear function* $f : (\{0,1\}^n)^k \to \{0,1\}^n$*, it holds that*

$$\Pr\left[f(m_1, \ldots, m_k) \leftarrow \mathsf{Dec}\left(\mathsf{sk}, \mathsf{LinFunc}(\mathsf{pk}, f, ct_1, \ldots, ct_k)\right)\right] = 1,$$

where $\{ct_i \leftarrow \mathsf{Enc}(\mathsf{sk}, m_i; r_i)\}_{i\in[l]}$.

Use $(\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Refresh}, \mathsf{Linfunc})$ **to design an IND-CPA secure public-key encryption scheme** $(\mathsf{KeyGen}_{\mathsf{PKE}}, \mathsf{Enc}_{\mathsf{PKE}}, \mathsf{Dec}_{\mathsf{PKE}})$**. Also prove IND-CPA security of your public key encryption scheme.**

4 Non-Interactive MPC

(10 Points) Let Alice and Bob be two parties with inputs $a \in \mathbb{Z}_q$ and $b \in \mathbb{Z}_q$, respectively. They wish to check if their inputs are equal, i.e., whether $a = b$. They want to do this while making sure that they do not learn any other information about the other party's input. In other words, if $a \neq b$, then Alice should not learn b and Bob should not learn a.

Let $\mathbb{G}$ be a cyclic group of prime order q with generator g. They run the following protocol:

- Alice samples a random value $r \leftarrow\$ \mathbb{Z}_q$. It then computes $X = g^r$ and $Y = g^{ar}$. It sends (X, Y) to Bob.
- Bob computes X^b. It outputs 1 if $X^b = Y$, and 0 otherwise.

Explain why this protocol is not secure against semi-honest Bob.