Fundamentals of Cryptography: Problem Set 11

Due Wednesday Dec 18, 3PM

Collaboration is permitted (and encouraged); however, you must write up your own solutions and acknowledge your collaborators.

Problem 0 Check lecture 17, 18, 19, 20 of course 6.875 in `mit6875.org`.

Problem 1 (7pt) Threshold Secret Sharing Lower Bound t-out-of-n secret sharing is a randomized algorithm $\mathsf{Share} : \mathcal{X} \times \mathcal{R} \to \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n$, where $\mathcal{X}$ is the secret space, $\mathcal{R}$ is the randomness space, and $\mathcal{S}_i$ the i-th share space.

(Prefect) Correctness For any subset $T \subseteq [n]$ such that $|T| \geq t$, there is a recovering algorithm Rec_T, such that for any $x \in \mathcal{X}, r \in \mathcal{R}$, let $(s_1, \ldots, s_n) = \mathsf{Share}(x, r)$, we have $\mathsf{Rec}_T((s_i)_{i \in T}) = x$.

(Prefect) Privacy For any subset $T \subseteq [n]$ such that $|T| < t$, for any $x, x' \in \mathcal{X}$, the two distribution

$$\mathsf{Share}_T(x, r), \qquad \mathsf{Share}_T(x', r)$$

are identical, where the randomness comes from $r \leftarrow \mathcal{R}$. Here $\mathsf{Share}_T(x, r)$ consists of all the i-th coordinate of $\mathsf{Share}(x, r)$ for $i \in T$.

As we mentioned in the class, Shamir secret sharing is a secure threshold secret sharing. The t-out-of-n Shamir secret sharing is as follows. Let $\mathbb{F}$ be a finite field of size at least $n + 1$. The sharing algorithm, given secret s, samples a random degree-at-most-$(t-1)$ polynomial p over $\mathbb{F}$ such that $p(0) = x$, and output $p(1), \ldots, p(n)$ as the shares.

In this problem, we show that the share size of Shamir secret sharing is close to optimal, even if the secret is only 1-bit (i.e., $\mathcal{X} = \{0, 1\}$).

Part A. Let $\mathsf{Share} : \mathcal{X} \times \mathcal{R} \to \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n$ be a 2-out-of-n secret sharing. Show that $\sum_{i=1}^{n} \log |\mathcal{S}_i| \geq \Omega(n \log n)$.

Hint: Show that $\sum_{i=1}^{n} \frac{1}{|\mathcal{S}_i|} \leq 1$.

Part B. Let $\mathsf{Share} : \mathcal{X} \times \mathcal{R} \to \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n$ be a t-out-of-n secret sharing, for some $t > 1$. Show that $\sum_i \log |\mathcal{S}_i| \geq \Omega((n - t) \log(n - t))$.

Remark: We also know $\sum_i \log |\mathcal{S}_i| \geq \Omega(t \log t)$ if $t < n$ [Bogdanov-Guo-Komargodski 2016]. Thus $\sum_i \log |\mathcal{S}_i| \geq \Omega(n \log n)$ unless $t = 1$ or n.

Problem 2 (5pt) Ramp Secret Sharing In the previous problem, we show that for t-out-of-n threshold secret sharing, the (average) sharing size is at least $\Omega(\log n)$, even if the secret is only 1 bit. We care about the ratio between the (largest) sharing size and secret length. For 1-bit secret, this ratio is $\Theta(\log n)$. But for longer secret, this ratio can be improved. If the secret is at least $\log n$ bit long, then every sharing can be as long as the secret.

The ratio between the (largest) sharing size and secret length, when the secret is sufficiently long, is called the *information ratio* of the secret sharing. So the information ratio of threshold secret sharing is 1. One can argue that 1 is a nature lower bound of information ratio.

However, for ramp secret sharing, as we are going to show, the information ratio can be smaller than 1. The (k, ℓ, n)-ramp secret sharing is a randomized algorithm $\mathsf{Share} : \mathcal{X} \times \mathcal{R} \to \mathcal{S}_1 \times \mathcal{S}_2 \times \cdots \times \mathcal{S}_n$, where $\mathcal{X}$ is the secret space, $\mathcal{R}$ is the randomness space, and $\mathcal{S}_i$ the i-th share space.

(Prefect) Correctness For any subset $T \subseteq [n]$ such that $|T| \geq \ell$, there is a recovering algorithm Rec_T, such that for any $x \in \mathcal{X}, r \in \mathcal{R}$, let $(s_1, \ldots, s_n) = \mathsf{Share}(x, r)$, we have $\mathsf{Rec}_T((s_i)_{i \in T}) = x$.

(Prefect) Privacy For any subset $T \subseteq [n]$ such that $|T| \leq k$, for any $x, x' \in \mathcal{X}$, the two distribution

$$\mathsf{Share}_T(x, r), \qquad \mathsf{Share}_T(x', r)$$

are identical for random $r \in \mathcal{R}$. Here $\mathsf{Share}_T(x, r)$ consists of all the i-th coordinate of $\mathsf{Share}(x, r)$ for $i \in T$.

Apparently, k, ℓ should satisfy $k < \ell \leq n$. Note that, the t-out-of-n Shamir secret sharing is $(t-1, t, n)$-ramp secret sharing.

Your task is to construct a (k, ℓ, n)-ramp secret sharing scheme for any $k < \ell \leq n$, such that its information ratio is bounded by

$$\frac{\max_i \log |\mathcal{S}_i|}{\log |\mathcal{X}|} = O\Big(\frac{1}{\ell - k}\Big).$$

State your construction and prove its correctness and privacy.

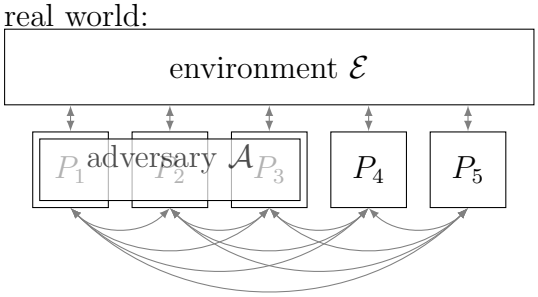
Problem 3 (6pt) Perfect security against semi-honest adversary means the view of the adversary can be perfectly simulated given the corrupted party's input and output. Show that *no* 2-party computation protocol computing the AND function is perfectly secure against semi-honest adversaries.
Remark: The claim can be generalized to statistically secure protocols.

Problem 4 (6pt) PKO Implies Security with Selective Abort In this problem, we consider a weaker security notion "privacy with knowledge of outputs (PKO)". In the new security notion, the adversary learns no extra information compare to an honest execution, but the adversary can choose an incorrect output for each honest party.

Assume that, any $f \in \mathbf{P/poly}$ has an efficient multi-party computation protocol which achieves (computational) PKO security against up to t corruptions. Show that, for any $f \in \mathbf{P/poly}$, there is also an efficient multi-party computation protocol which achieves (computational) security with selective abort against up to t corruptions.

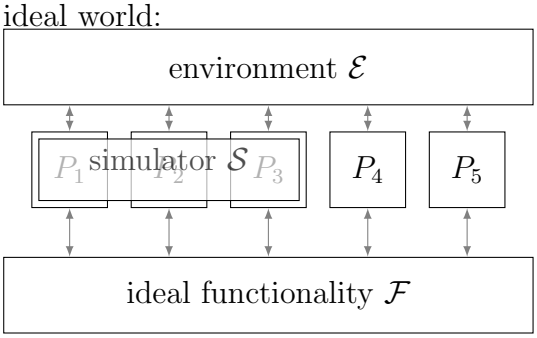
Malicious Security Definitions An n-party MPC protocol computing a function f is computationally (resp. perfectly/statistically) secure against up to t active corruptions, if for any p.p.t. (resp. unbounded) environment $\mathcal{E}$, adversary $\mathcal{A}$, for any set $S \subseteq [n]$ of at most t corrupted parties, there exists a p.p.t. (resp. unbounded) simulator $\mathcal{S}$, such that the view of the environment in the real world and the ideal world are computationally (resp. perfectly/statistically) indistinguishable.

real world:



Honest P_i receives input x_i from $\mathcal{E}$, executes the protocol, sends the protocol's output y_i to $\mathcal{E}$.
All corrupted parties are controlled by the adversary. They may interacts arbitrarily with $\mathcal{E}$ and honest parties.

ideal world:



Honest P_i receives input x_i from $\mathcal{E}$, forwards x_i to $\mathcal{F}$, receives y_i from $\mathcal{F}$ sends y_i to $\mathcal{E}$.
All corrupted parties are controlled by the simulator. Each corrupted party P_i sends x_i to $\mathcal{F}$ and receives y_i from $\mathcal{F}$. They may interacts arbitrarily with $\mathcal{E}$.

There are several different level of security definitions, depending on how the ideal functionality is define.

Full Security = Guaranteed Output Delivery (GOD) Security: Upon receiving x_i from every party P_i, locally compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, send y_i to P_i.

Security with Abort: Upon receiving x_i from every party P_i, locally compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, send y_i to P_i if P_i is corrupted.

Wait for an signal $b_{\text{abort}} \in \{0, 1\}$ from $\mathcal{S}$. If $b_{\text{abort}} = 1$, send $\perp$ to all honest parties; otherwise send y_i to P_i.

Security with Selective Abort: Upon receiving x_i from every party P_i, locally compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, send y_i to P_i if P_i is corrupted.

For each honest P_i, wait for signals $b_{\text{abort,i}} \in \{0, 1\}$ from $\mathcal{S}$. If $b_{\text{abort,i}} = 1$, send $\perp$ to P_i; otherwise send y_i to P_i.

Privacy with Knowledge of Output (PKO) Security: Upon receiving x_i from every party P_i, locally compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, send y_i to P_i if P_i is corrupted.

For each honest P_i, wait for $\hat{y}_i \in \{0, 1\}$ from $\mathcal{S}$. If $\hat{y}_i = \top$, send y_i to P_i; otherwise send $\hat{y}_i$ to P_i.