Fundamentals of Cryptography: Problem Set 11

Due Wednesday Dec 24, 3PM

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

If a problem has **0pt**, it will not be graded.

Problem 1 (5pt) 1-out-of-t Oblivious Transfer 1-out-of-t OT is a natural generalization of the standard oblivious transfer. In 1-out-of-t OT, the sender is given t equal-length messages $m_1, \dots, m_t \in \{0,1\}^\ell$, the receiver is given an index $x \in \{1, \dots, t\}$, the protocol let the receiver learn m_x, without revealing any other information.

Correctness After the protocol, the receiver always output m_x.

Receiver's Security against Semi-honest Sender The sender doesn't learn anything about the receiver's index. Let $\mathsf{View}_S((m_1, \dots, m_t), x)$ denote the view of the sender when the sender is given t messages $m_1, \dots, m_t$, the receiver is given an index $x \in \{1, \dots, t\}$. There exists an efficient simulator Sim_S such that

$$\mathsf{View}_S((m_1, \dots, m_t), x) \approx \mathsf{Sim}_S(m_1, \dots, m_t)\,.$$

The security is perfect, statistical, or computational, if the above two distributions are perfectly, statistically, or computationally indistinguishable.

Sender's Security against Semi-honest Receiver The receiver doesn't learn anything about the other messages of the sender. Let $\mathsf{View}_R((m_1, \dots, m_t), x)$ denote the view of the receiver when the sender is given t messages $m_1, \dots, m_t$, the receiver is given an index $x \in \{1, \dots, t\}$. There exists an efficient simulator Sim_R such that

$$\mathsf{View}_R((m_1, \dots, m_t), x) \approx \mathsf{Sim}_R(m_x, x)\,.$$

The security is perfect, statistical, or computational, if the above two distributions are perfectly, statistically, or computationally indistinguishable.

Show how to construct a (semi-honest) 1-out-of-t OT protocol based on a given 1-out-of-2 OT (i.e., the standard OT) protocol. You can use $\mathsf{OT.Sim}_S$ and $\mathsf{OT.Sim}_R$ to denote the simulators of the 1-out-of-2 OT protocol.

Problem 2 (6pt) Multi-key MAC Your task is to construct a *multi-key MAC* scheme. Such a scheme is a tuple of three algorithms $(\mathsf{Gen}, \mathsf{MAC}, \mathsf{Verify})$.

$\mathsf{Gen}(1^n, i)$ generates a key k.

$\mathsf{MAC}((k_1, \dots, k_n), m)$ takes n keys and a message, outputs a tag t.

$\mathsf{Verify}(i, k, m, t)$ takes an index, a key, a message and a tag, outputs a bit indicating acceptance or rejection.

A multi-key MAC scheme should be correct and unforgeable: Correctness means any PPT adversary wins the following correctness game with at most negl(n) probability.

> The adversary is given 1^n and outputs m. For each $1 \leq i \leq n$, key k_i is generated by $\mathsf{Gen}(1^n, i)$. Compute a tag $t \leftarrow \mathsf{MAC}((k_1, \ldots, k_n), m)$. The adversary wins if $\exists i, \mathsf{Verify}(i, k_i, m, t)$ rejects.

Unforgeability means any PPT adversary $\mathcal{A}$ wins the following adaptive chosen message attack game with at most negl(n) probability.

1. Keys $k_1, \ldots, k_n$ are independently generated by $\mathsf{Gen}(1^n, 1), \ldots, \mathsf{Gen}(1^n, n)$.
2. The adversary $\mathcal{A}$ is given input 1^n and oracle access to $\mathsf{MAC}((k_1, \ldots, k_n), \cdot)$. That is, if the adversary chooses m_j in its j-th oracle query, the adversary will receive tag $t_j \leftarrow \mathsf{MAC}((k_1, \ldots, k_n), m_j)$. Eventually, the adversary outputs (m, t, i).
3. The adversary wins if 1) $\mathsf{Verify}(i, k_i, m, t)$ accepts, and 2) $(m, t) \neq (m_j, t_j)$ for all j.

Part A. Your task is to construct such a multi-key MAC scheme and briefly prove its security. As for computational assumption(s), you can use OWF, OWP, PRG, PRF, PRP, (keyed or keyless) CRHF in your construction.

Part B. If you cannot solve part A, you may instead construct a *one-time* multi-key MAC scheme. The definition is the same, except that the adversary $\mathcal{A}$ can only make one oracle query in the unforgeability game.

(For one-time security, computational assumptions are unnecessary, though you can still use them in your construction.)

Problem 3 (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.

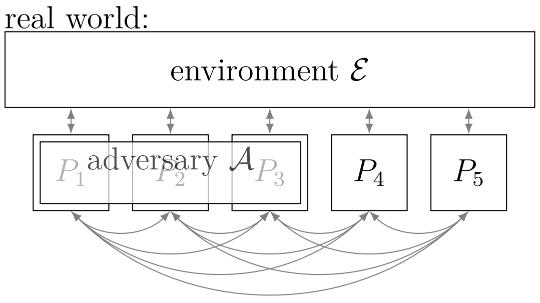
Assume that, any $f \in \mathbf{P/poly}$ has an efficient multi-party computation protocol which achieves (computational) PKO security against up to t corruptions. Show that, for any $f \in \mathbf{P/poly}$, there is also an efficient multi-party computation protocol which achieves (computational) security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

Remark: In the above statement, "security with selective abort" can be easily strengthed to "security with abort", using one-time multi-key MAC.

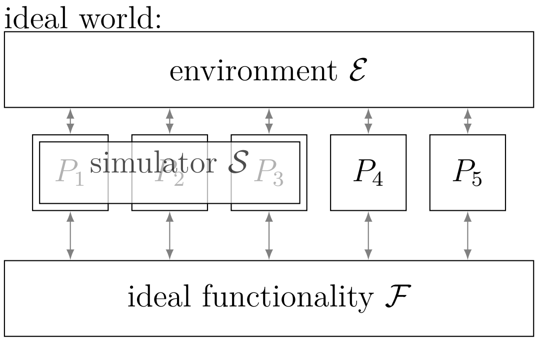
Malicious Security Definitions An n-party MPC protocol computing a function f is computationally (resp. perfectly/statistically) secure against up to t active corruptions, if for any p.p.t. (resp. unbounded) environment $\mathcal{E}$, adversary $\mathcal{A}$, for any set $S \subseteq [n]$ of at most 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.

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, compute $(y_1, \ldots, y_n) = f(x_1, \ldots, x_n)$, send y_i to P_i.



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.



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 $\mathcal{E}$.

Security with Abort: Upon receiving x_i from every party P_i, 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 the adversary (= simulator $\mathcal{S}$). 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, 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 the adversary (= simulator $\mathcal{S}$). 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, 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 message $\hat{y}_i \in \{0, 1\}$ from the adversary (= simulator $\mathcal{S}$). If $\hat{y}_i = \top$, send y_i to P_i; otherwise send $\hat{y}_i$ to P_i.