CS 65500
Advanced Cryptography

Lecture 4: Oblivious Transfer II

Instructor: Aarushi Goel

Spring 2025

Agenda

→ Oblivious Transfer

- → Recall Construction
- → Security Proof.

Semi-Honest Secure Two-Party Computation

Definition: A protocol π securely computes a function f in the semi-honest model, if $\exists$ a pair of two n.u. PPT simulator algorithms S_A and S_B, such that for every security parameter k, and $\forall$ inputs $x, y \in \{0,1\}^k$, it holds that:

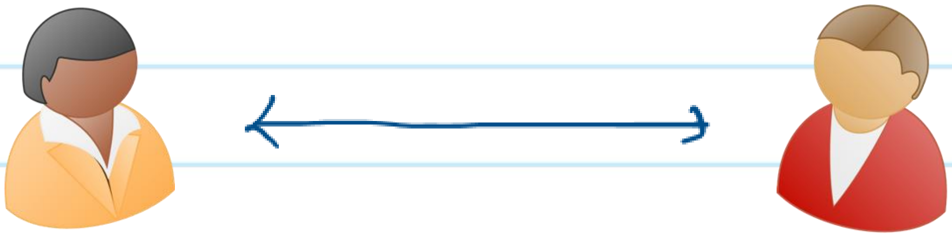
$$\{S_A(x, f(x,y)), f(x,y)\} \approx \{view_A(e), out_B(e)\}$$

$$\{S_B(y, f(x,y)), f(x,y)\} \approx \{view_B(e), out_A(e)\}$$

where $e \leftarrow [A(x) \leftrightarrow B(y)]\}$

Oblivious Transfer (OT)

Consider the following functionality :



	Alice	Bob
Input :	(a_0, a_1)	b
Output :	$\perp$	a_b

Security: Alice doesn't learn b.

Bob doesn't learn a_{1-b}

Constructing Oblivious Transfer

Building Block I

Hardcore Predicate: Hard core bit cannot be predicted with probability $> 1/2 + negl(K)$, even given the output of a one-way function.

Definition: A predicate $h: \{0,1\}^* \rightarrow \{0,1\}$ is a hardcore predicate for a OWF $f(.)$, if it is efficiently computable given x, $\exists$ a negligible function $\nu(.)$, s.t., $\forall$ n.u. PPT adv A, & $\forall$ security parameters K,

$$\Pr\left[A(1^K, f(x)) = h(x) \,;\, x \xleftarrow{\$} \{0,1\}^K\right] \leq \frac{1}{2} + \nu(K)$$

Constructing Oblivious Transfer: Building Block II

Trapdoor One-way Permutations: A collection of permutations is a family of permutations $F = \{f_i : D_i \to R_i\}_{i \in I}$ satisfying the following properties:

- Sampling Function: $\exists$ a PPT Gen, s.t. $Gen(1^k) \to (i \in I, t)$
- Sampling from Domain: $\exists$ a PPT algorithm that on input i outputs a uniformly random element of D_i
- Evaluation: $\exists$ a PPT algorithm that on input $i, x \in D_i$, outputs $f_i(x)$.
- Inversion with trapdoor: $\exists$ a PPT algorithm Inv s.t.

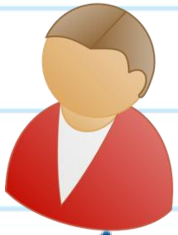
$$Inv(i, t, y) \longrightarrow f_i^{-1}(y)$$

- Hard to invert: $\forall$ n.u. PPT adv A, $\exists$ a negl fun $\nu(.)$, s.t.,

$$Pr[f_i(A(1^k, i, y)) = x \; ; \; i \leftarrow Gen(1^k), \; x \leftarrow D_i, \; y \leftarrow f_i(x)] \leqslant \nu(k)$$

Construction of Oblivious Transfer.

Alice



Bob

Input: (a_0, a_1) — b

Protocol: $(f_i, f_i^{-1}) \leftarrow Gen(1^k)$ $\xrightarrow{f_i}$ $x \xleftarrow{\$} \{0,1\}^k$, $y_{1-b} \xleftarrow{\$} \{0,1\}^k$

$\xleftarrow{(y_0, y_1)}$ $y_b = f_i(x)$

$\forall j \in \{0,1\}$

$z_j = h(f_i^{-1}(y_j)) \oplus a_j$

$\xrightarrow{(z_0, z_1)}$

Output $h(x) \oplus z_b$

This is a Semi-Honest Oblivious Tranfer

→ Security against Alice: Both y_0 & y_1 are uniformly distributed and therefore independent of b.

→ Security against Bob: If Bob could learn a_{1-b}, then he would be able to predict the hardcore predicate.

Does this construction remain secure if either Alice or Bob were malicious?

Security Proof

Simulator $S_A((a_0, a_1), \perp)$

1. Fix a random tape r_A for Alice. Use this to sample
   $(f_i, f_i^{-1}) \leftarrow Gen(1^K)$

2. Choose two random strings $y_0, y_1 \leftarrow \{0,1\}^K$ as Bob's msg

3. $\forall j \in \{0,1\}$, compute $z_j = h(f_i^{-1}(y_j)) \oplus a_j$ to obtain the third msg (z_0, z_1)

4. Stop & output $\perp$

Security Proof

Claim: The following two distributions are identical:

$$\{S_A((a_0,a_1),\perp), a_b\} \text{ and }$$

$$\{\mathrm{View}_A(e), \mathrm{Out}_B(e) ; e \leftarrow [A(a_0,a_1) \longleftrightarrow B(b)]\}$$

Proof Idea: The only difference between S_A and the real execution is how y_0, y_1 are computed. However, since f_i is a permutation, y_0, y_1 are identically distributed in both cases.

Security Proof

<u>Simulator</u> $S_B(b, a_b)$:

1. Sample f_i

2. Choose a random tape r_B for B. Use that to compute $x \xleftarrow{\$} \{0,1\}^K$, $y_b = f_i(x)$, $y_{1-b} \xleftarrow{\$} \{0,1\}^K$

3. Compute $z_b = h(x) \oplus a_b$, $z_{1-b} \xleftarrow{\$} \{0,1\}$

4 Output (z_0, z_1) as the third msg and stop.

Security Proof

Claim: The following two distributions are identical:

$$\{ S_B(b, a_b), \perp \}$$

$$\{ View_B(e), Out_A(e) ; e \leftarrow [A(a_0, a_1) \longleftrightarrow B(b)] \}$$

Proof Idea: The only difference between S_B and the real execution is how z_0, z_1 are computed. However, since $h(f_i^{-1}(y_{1-b}))$ is computationally indistinguishable from random (even given y_{1-b}), this change is computationally indistinguishable.

Security Proof

To Prove: $\{ S_B(b, a_b), \perp \}$

$\approx_c \{ \mathrm{View}_B(e), \mathrm{Out}_A(e) ; e \leftarrow [A(a_0, a_1) \longleftrightarrow B(b)] \}$

$$\left\{ f_i \xleftarrow{\$} Gen(1^K), x \xleftarrow{\$} \{0,1\}^K, y_b = f_i(x), y_{1-b} \xleftarrow{\$} \{0,1\}^K, z_b = h(x) \oplus a_b, z_{1-b} \xleftarrow{\$} \{0,1\} \right\}$$

$$\left\{ f_i \xleftarrow{\$} Gen(1^K), x \xleftarrow{\$} \{0,1\}^K, y_b = f_i(x), y_{1-b} \xleftarrow{\$} \{0,1\}^K, z_b = h(f_i^{-1}(y_b)) \oplus a_b, z_{1-b} = h(f_i^{-1}(y_{1-b})) \oplus a_{1-b} \right\}$$

Security Proof

$H_1: \{ f_i \xleftarrow{\$} Gen(1^K),\ x \xleftarrow{\$} \{0,1\}^K,\ y_b = f_i(x),\ y_{1-b} \xleftarrow{\$} \{0,1\}^K,$
$z_b = h(x) \oplus a_b,\ z_{1-b} \xleftarrow{\$} \{0,1\} \}$

$H_2: \{ f_i \xleftarrow{\$} Gen(1^K),\ x \xleftarrow{\$} \{0,1\}^K,\ y_b = f_i(x),\ y_{1-b} \xleftarrow{\$} \{0,1\}^K,$
$z_b = h(x) \oplus a_b,\ z \xleftarrow{\$} \{0,1\},\ z_{1-b} = z \oplus a_{1-b} \}$

$H_3: \{ f_i \xleftarrow{\$} Gen(1^K),\ x \xleftarrow{\$} \{0,1\}^K,\ y_b = f_i(x),\ y_{1-b} \xleftarrow{\$} \{0,1\}^K,$
$z_b = h(f_i^{-1}(y_b)) \oplus a_b,\ z_{1-b} = h(f_i^{-1}(y_{1-b})) \oplus a_{1-b} \}$

$H_1 \equiv H_2$: Security of one-time pad

Security Proof

We want to show that $\forall b, a_b, a_{1-b} \in \{0,1\}^3$, the following distributions are computionally indistinguishable:

$$H_2: \left\{ f_i \xleftarrow{\$} Gen(1^K),\ x \xleftarrow{\$} \{0,1\}^K,\ y_b = f_i(x),\ y_{1-b} \xleftarrow{\$} \{0,1\}^K,\ z_b = h(x) \oplus a_b,\ z \xleftarrow{\$} \{0,1\},\ z_{1-b} = z \oplus a_{1-b} \right\}$$

$$H_3: \left\{ f_i \xleftarrow{\$} Gen(1^K),\ x \xleftarrow{\$} \{0,1\}^K,\ y_b = f_i(x),\ y_{1-b} \xleftarrow{\$} \{0,1\}^K,\ z_b = h(f_i^{-1}(y_b)) \oplus a_b,\ z_{1-b} = h(f_i^{-1}(y_{1-b})) \oplus a_{1-b} \right\}$$

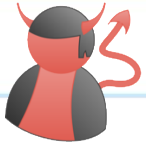
What does the security game for this indistinguishability look like?

Proof by Reduction

Let us assume for the sake of contradiction that $\exists$ adv A, who can distinguish b/w H_2 & H_3 with non-negl advantage ν. We will use this adv to design another adv B, who can break security of hard core predicates.

Security game for HCP corresponding to trapdoor OWP:

Ch



Adv

$\alpha \xleftarrow{\$} \{0,1\}$, $(f_i, f_i^{-1}) \leftarrow Gen(1^k)$,

$x \xleftarrow{\$} \{0,1\}^k$, $y = f_i(x)$

if $\alpha = 0$: $z \xleftarrow{\$} \{0,1\}$

if $\alpha = 1$: $z = h(x)$

$\xrightarrow{f_i, y, z}$

$\xleftarrow{\alpha'}$

Adv wins if $\alpha' = \alpha$

Proof by Reduction

Ch

$\alpha \xleftarrow{\$} \{0,1\}$

$(f_i, f_i^{-1}) \leftarrow Gen(1^k)$

$y \xleftarrow{\$} \{0,1\}^k$

if $\alpha = 0$:

$\quad z \xleftarrow{\$} \{0,1\}$

if $\alpha = 1$:

$z = h(f_i^{-1}(y))$

$\xrightarrow{f_i,\, z,\, y}$

$\xleftarrow{\alpha''}$

B

$x \xleftarrow{\$} \{0,1\}^k$

$z_b = h(x) \oplus a_b$

$z_{1-b} = z \oplus a_{1-b}$

if $\alpha' = 2$: $\alpha'' = 0$

if $\alpha' = 3$: $\alpha'' = 1$

A

$\xleftarrow{b,\, a_0,\, a_1}$

$\xrightarrow{f_i,\, x,\, y,\, z_0,\, z_1}$

$\xleftarrow{\alpha'}$