CS 555 Lecture 19

Oblivious Transfer

November 4, 2025

Protocol Overview

Parties: A and B
Messages: M_0 and M_1.
Goal: B wants M_0 but doesn't want A to know which it wants. A doesn't want B to know M_1.
Privacy for both sender and receiver.
Note: Privacy can rigorously be defined through simulation-based games.

Protocol Construction

1. Public Key Encryption: Given pk, it is hard to find sk.
2. B **samples:** (pk_0, sk_0); chooses random r and defines $pk_1 = r \oplus pk_0$. Send pk_0, pk_1.
3. A **encrypts:** m_0 using pk_0 and m_1 using pk_1.

Security Questions

A doesn't know which message B wants.
But what if B samples pk_0, sk_0? This protocol is secure in the semi-honest model (parties follow the protocol but are curious).

In a malicious model:
Idea: A chooses g^r and B creates

$$pk_0 = g^x, \quad pk_1 = g^{rrr}, \text{ or } pk_1 g^y = pk_0$$

Idea: B sends a ZKP: "I only know the sk of one of the pk_0, pk_1."
Since r is randomly chosen from a large field, the probability that two arbitrarily chosen public keys differ by a factor of g^r is very low.

Implications of Oblivious Transfer

The Billionaire Problem (Secure Computation):

A	B
private input x	private input y

Only $f(x, y)$ is revealed (and what it implies about x and y).

Yao's Two-Party Computation

Setup: Represent a computation as a circuit. We garble the circuit where A generates the garbled circuit and B evaluates the garbled circuit. The garbling is constructed in such a way that A learns nothing about B's input and B learns nothing even if it evaluates the circuit.

Truth Table Example for 1 Gate Circuit

x	y	z	x	y	z	x	y	z
0	0	0	ℓ_x^0	ℓ_y^0	ℓ_z^0	ℓ_x^0	ℓ_y^0	$\text{Enc}_{\ell_x^0,\ell_y^0}(\ell_z^0)$
0	1	0	ℓ_x^0	ℓ_y^1	ℓ_z^0	ℓ_x^0	ℓ_y^1	$\text{Enc}_{\ell_x^0,\ell_y^1}(\ell_z^0)$
1	0	0	ℓ_x^1	ℓ_y^0	ℓ_z^0	ℓ_x^1	ℓ_y^0	$\text{Enc}_{\ell_x^1,\ell_y^0}(\ell_z^0)$
1	1	1	ℓ_x^1	ℓ_y^1	ℓ_z^1	ℓ_x^1	ℓ_y^1	$\text{Enc}_{\ell_x^1,\ell_y^1}(\ell_z^1)$

The values $\ell_x^0, \ell_y^0, \ell_z^0, \ell_x^1, \ell_y^1, \ell_z^1$ are random strings.

Understanding Labels and Table Construction

x knows the labels and how the table is constructed. y knows the input and wants to learn z.

OT in Practice

Public key encryption is very expensive. If our circuit is a ML model, using public key encryption is infeasible.
Idea: Use public key encryption to create small number of OTs and use symmetric key encryption to extend to many OTs. This is called **OT extension**.