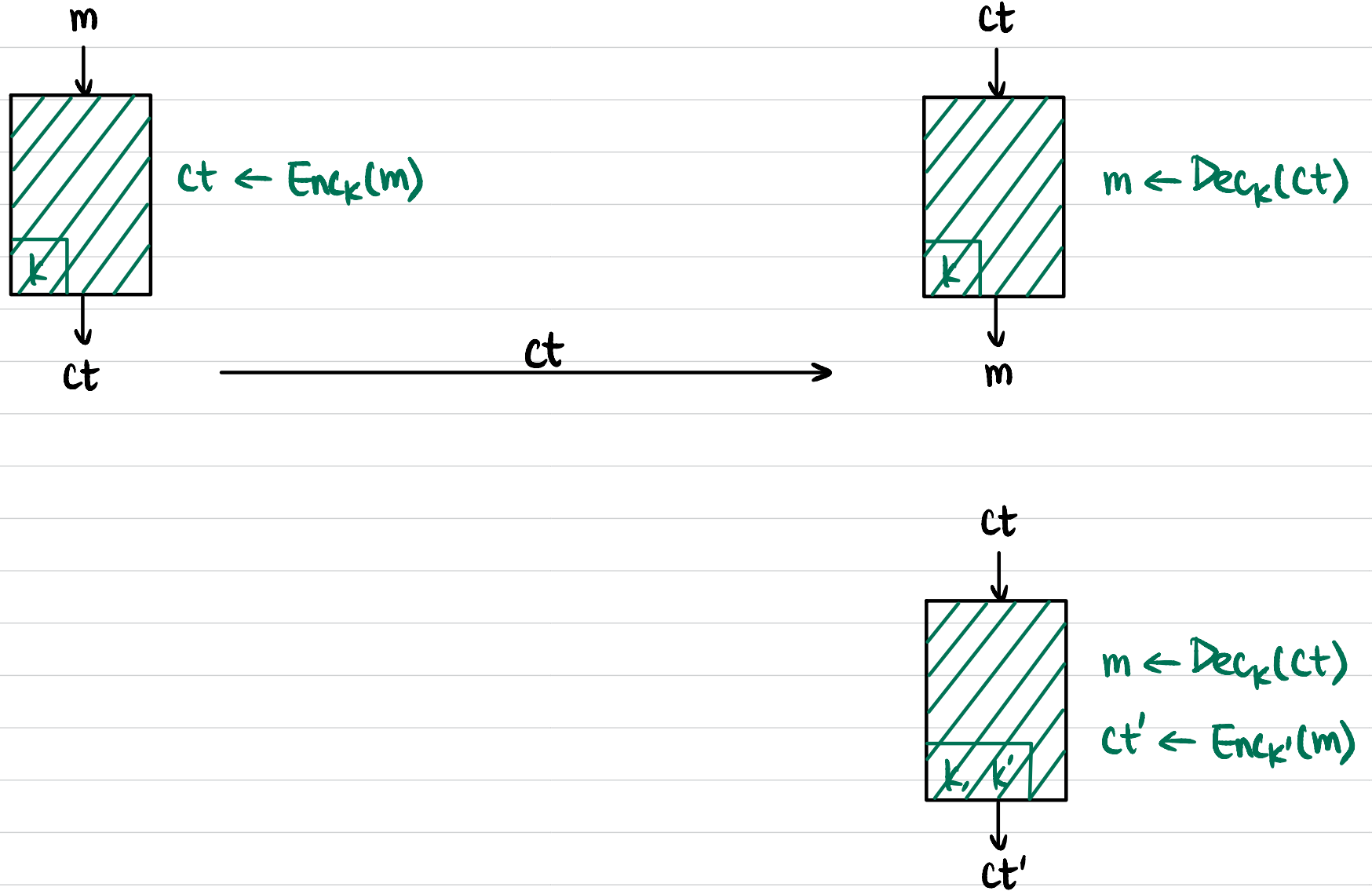


# CSCI 1515 Applied Cryptography

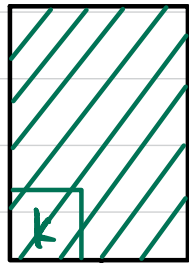
## This Lecture:

- Secure Hardware: HSM
- Introduction to Secure Multi-Party Computation
- Feasibility Results of MPC

# Hardware Secure Module (HSM)



# Key Agreement



Sample  $k_1, k_2, k_3$  s.t.  
 $k_1 \oplus k_2 \oplus k_3 = K$



$k_1$

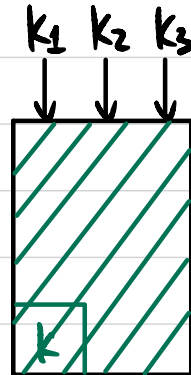
$k_2$

$k_3$

FedEx

UPS

USPS



$K := k_1 \oplus k_2 \oplus k_3$

# Secure Multi-Party Computation

Alice



$x$

Second date?

$$f(x, y) = x \wedge y$$

Bob



$y$

Who is richer?

$$f(x, y) = \begin{cases} 0 & \text{if } x > y \\ 1 & \text{otherwise} \end{cases}$$

Mutual friends?

$$f(x, y) = x \wedge y$$



# Secure Two-Party Computation (2PC)

Alice



$x$

Bob



$y$

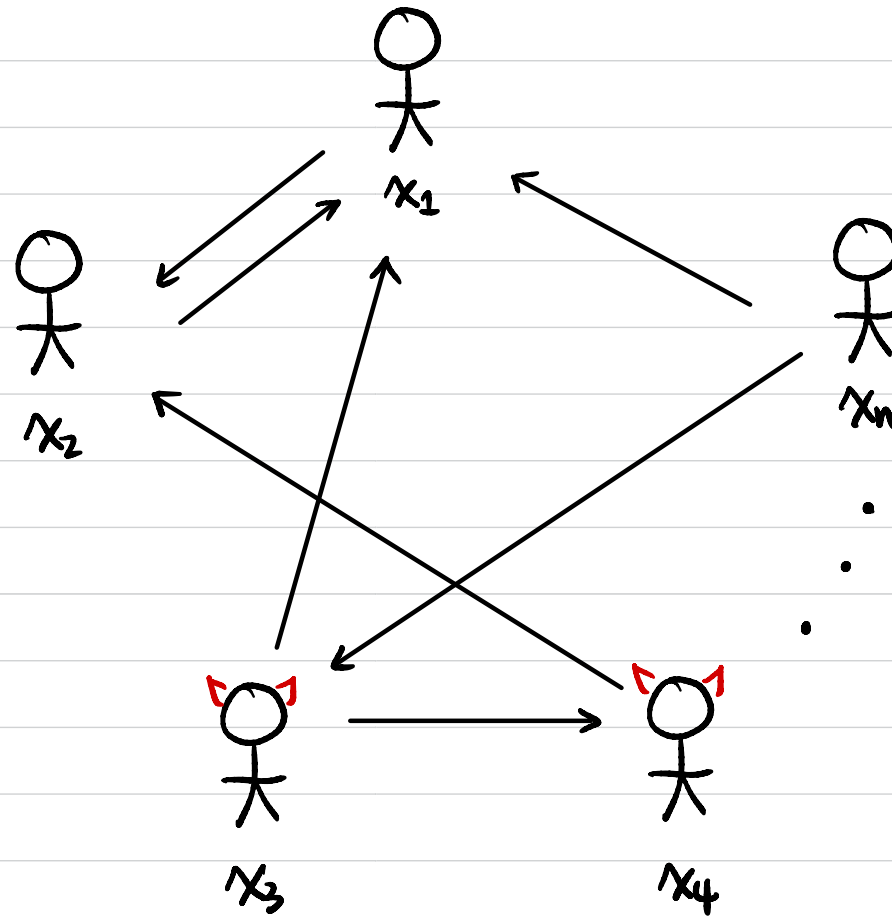


$$z = f(x, y)$$

## Applications:

- Password Breach Alert (Chrome / Firefox / Azure / iOS Keychain)
- Privacy-Preserving Contact Tracing for COVID-19 (Apple & Google)
- Ads Conversion Measurements / Personalized Advertising (Google / Meta)

# Secure Multi-Party Computation (MPC)




$$z = f(x_1, \dots, x_n)$$

# Secure Multi-Party Computation (MPC)

## Applications:

- Privacy-Preserving Inventory Matching (J.P. Morgan)
- Setup Ceremony to securely generate CRS (Zcash)
- Distributed Key Management (Unbound / Coinbase)
- Federated Learning (Google Keyboard Search Suggestion)
- Auctions (Danish sugar beet auction)
- Boston gender wage gap (Boston Women's Workforce Council)
- Study / Analysis on Medical Data
- Fraud / Money Laundering Detection (banks)

## Setting

- $n$  parties  $P_1, P_2, \dots, P_n$   
with private inputs  $x_1, x_2, \dots, x_n$
- Jointly compute  $f(x_1, x_2, \dots, x_n)$   
 *public*
- Communication:  
Authenticated secure point-to-point channels between each pair  $(P_i, P_j)$   
(Sometimes also assume broadcast channel)
- The adversary can "corrupt" a subset of the parties  
(e.g. at most  $t$  parties)

What properties do we want?

# General Security Properties

- **Correctness:** The function is computed correctly.
- **Privacy:** Only the output is revealed.
- **Independence of Inputs:** Parties cannot choose inputs depending on others' inputs.
- **Security with Abort:** Adversary may "abort" the protocol.  
 $t < n$  (preventing honest parties from receiving the output)
- **Fairness:** If one party receives output, then all receive output.  $t < \frac{n}{2}$
- **Guaranteed Output Delivery (GOD):** Honest parties always receive output.  $t < \frac{n}{3}$

# Adversary's Power

## Allowed adversarial behavior:

- Semi-honest / passive / honest-but-curious:  
Follow the protocol description honestly,  
but try to extract more information by inspecting transcript.
- Malicious / active:  
Can deviate arbitrarily from the protocol description.

## Adversary's Computing Power:

- Unbounded computing power  $\Rightarrow$  Information-Theoretic (IT) Security
- PPT bounded  $\Rightarrow$  Computational Security

# Feasibility Results

## Computational Security:

Semi-honest Oblivious Transfer (OT)



semi-honest MPC for any function with  $t < n$

# corrupted parties  
↙



malicious MPC for any function with  $t < n$  (security w/ abort)

## Information-Theoretic (IT) Security:

semi-honest/malicious MPC for any function with  $t < n/2$

(honest majority)

↑  
necessary

# Oblivious Transfer (OT)

Sender



Input:  $m_0, m_1 \in \{0, 1\}^l$

Output:  $\perp$



Receiver



Input:  $b \in \{0, 1\}$

Output:  $m_b$



Semi-honest OT  
(OT protocol that's secure  
against semi-honest adv.)

Yao's Garbled  
Circuit

Semi-honest ZPC  
for any function

Cut-and-choose  
with commitments

Malicious ZPC  
for any function

GMW

Semi-honest MPC  
for any function

GMW Compiler  
with ZKP

Malicious MPC  
for any function