CSE 539: Applied Cryptography
Week 12: Multi-party Computation

Ni Trieu (ASU)

Reading:

- https://web.engr.oregonstate.edu/~rosulekm/cryptabit/1-overview.pdf
- https://www.youtube.com/watch?v=FTxh908u9y8
- https://securecomputation.org/docs/ch1-introduction.pdf
- https://securecomputation.org/docs/ch2-definingmpc.pdf

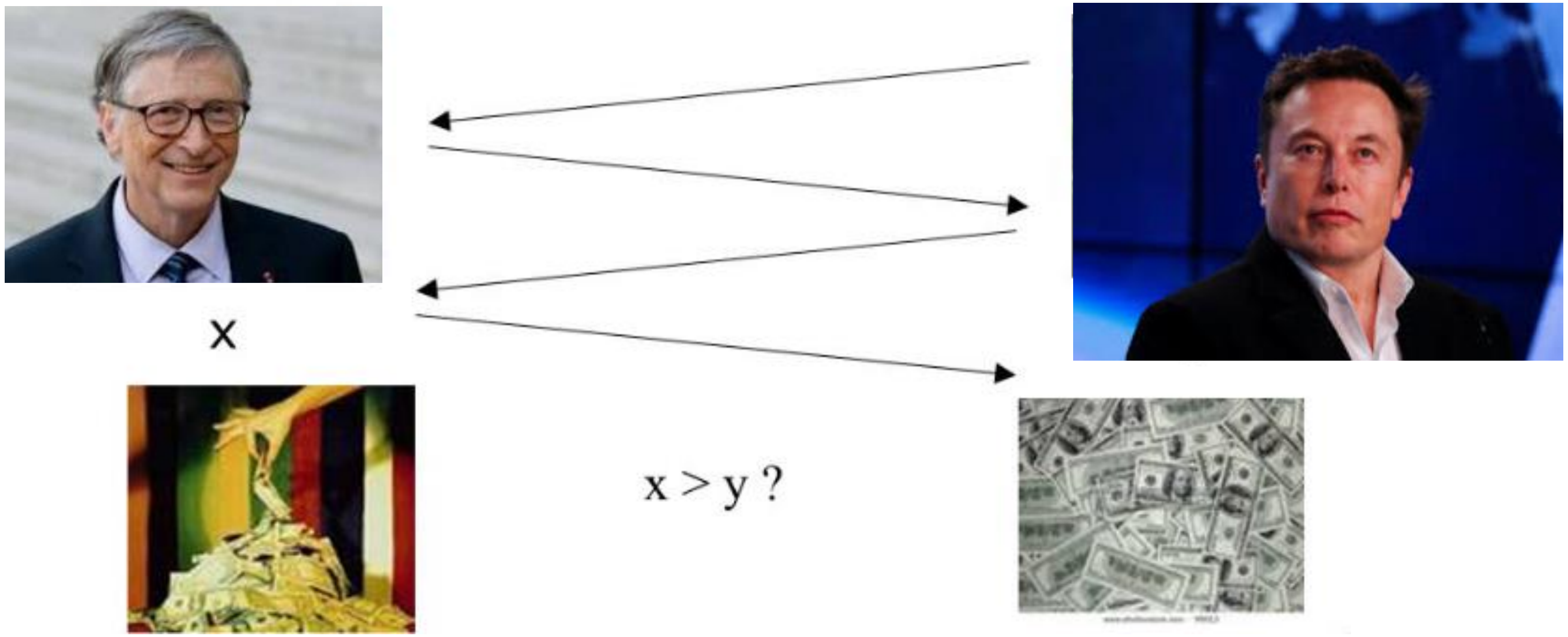
What is Secure Computation?

- Secure computation is a magic box

- Example:
  - Yao's Millionaires’ Problem
  - Private Matching
  - Secure Voting
  - Privacy-Preserving Machine Learning

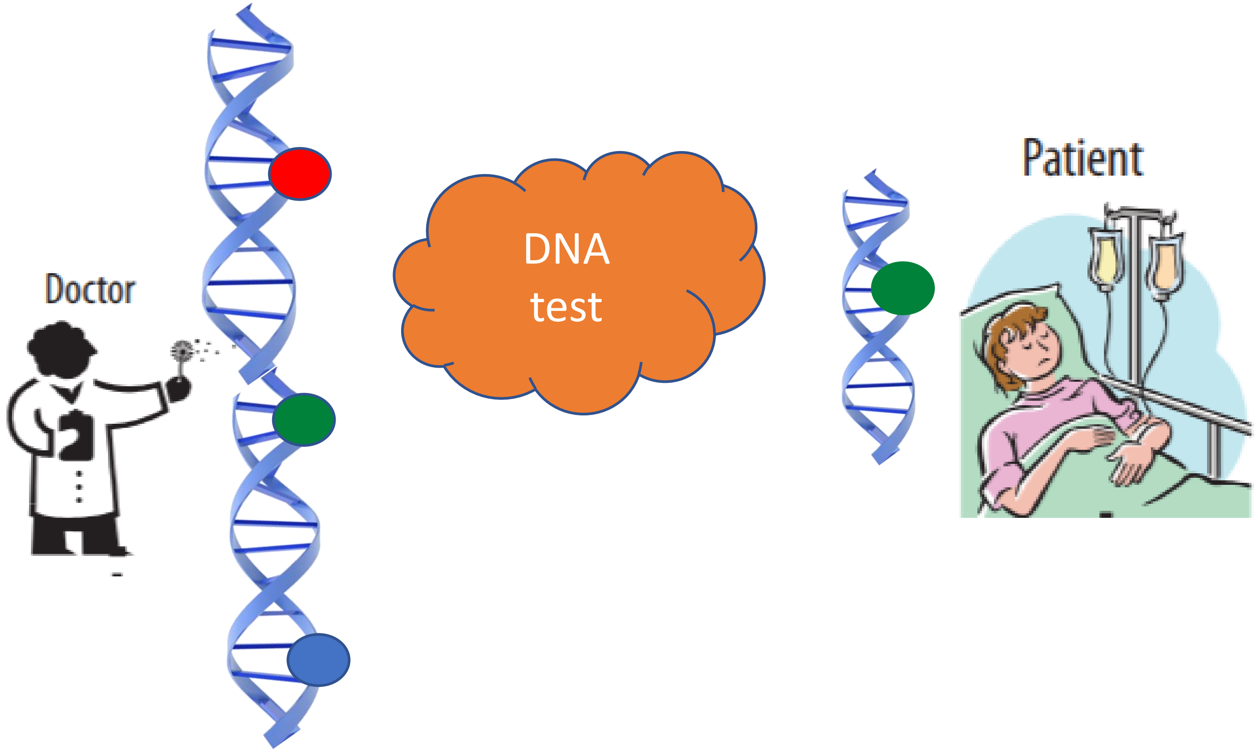
Yao's Millionaires' Problem

- How to determine who is richer while keeping their actual wealth private?



Private Matching

- How to do DNA testing without revealing the input data?



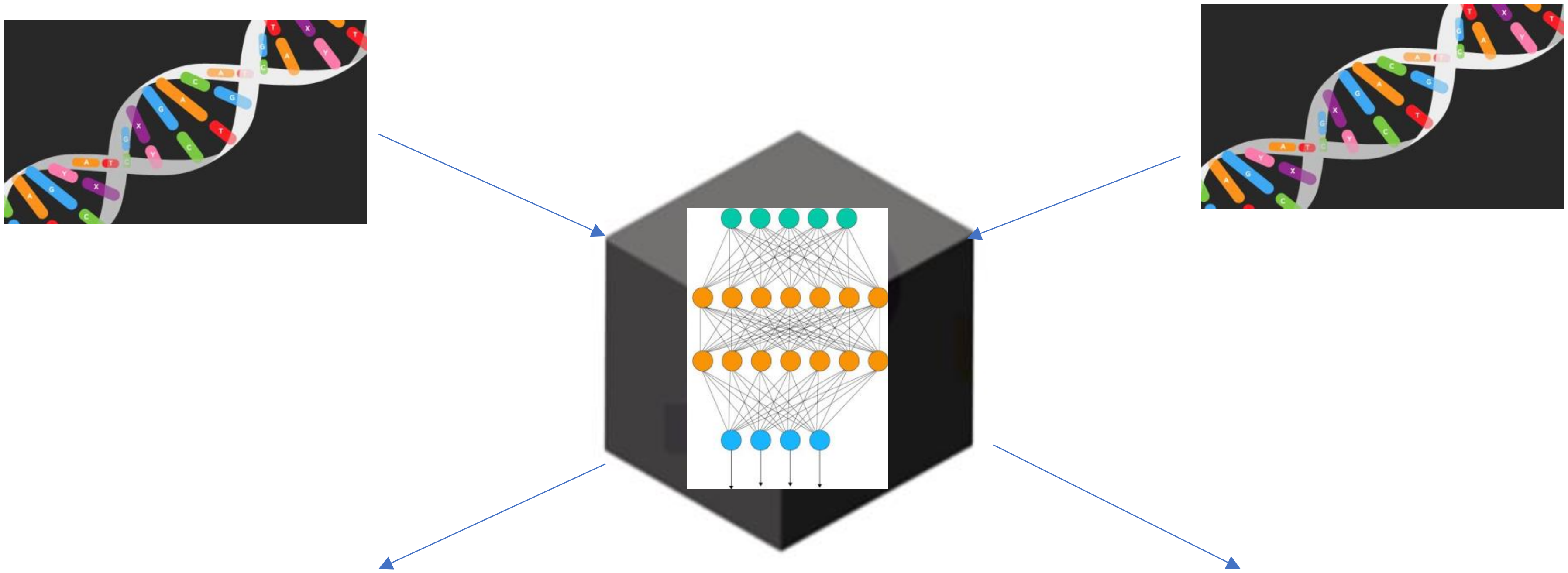
Secure Voting/Auction

- How to vote without revealing your vote?

Privacy-Preserving Machine Learning

- How to train a ML model while maintaining the privacy of each database (from different sources)

X



Y

$f(X, Y)$

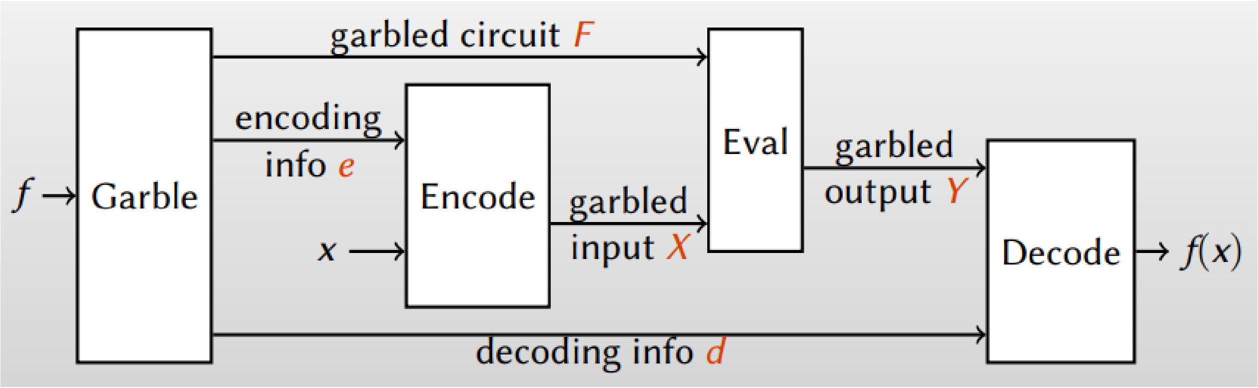
Learns nothing about Y

$f(X, Y)$

Learns nothing about X

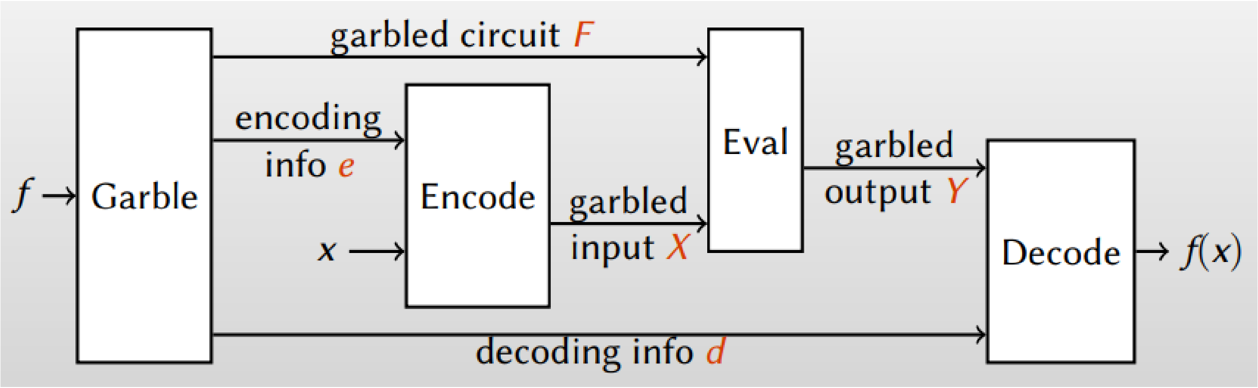
How to do Secure Computation?

- Multi-party Computation (Garbled Circuit or Secret Sharing)
  - Present the computation function f as a circuit
  - Evaluate the circuit via garbling, encoding, decoding “Obviously”

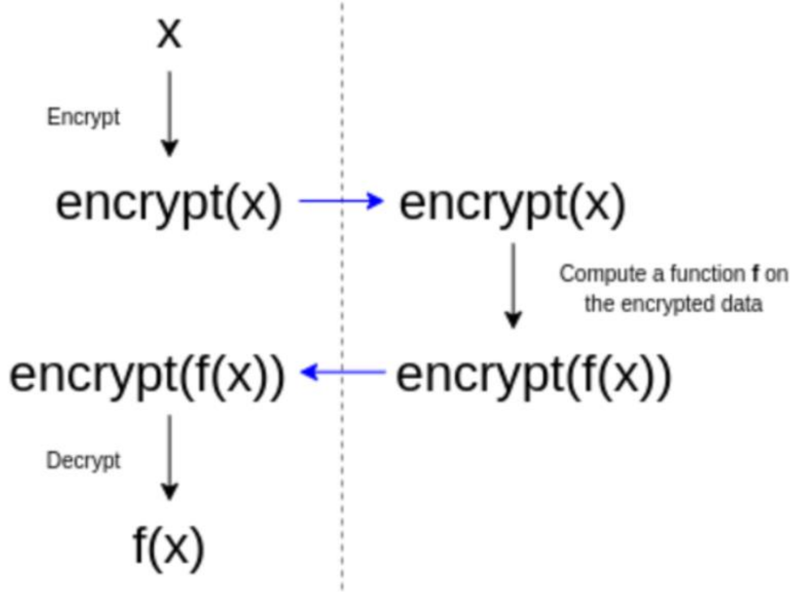


How to do Secure Computation?

- Multi-party Computation (Garbled Circuit or Secret Sharing)
  - Present the computation function f as a circuit
  - Evaluate the circuit via garbling, encoding, decoding "Obviously"

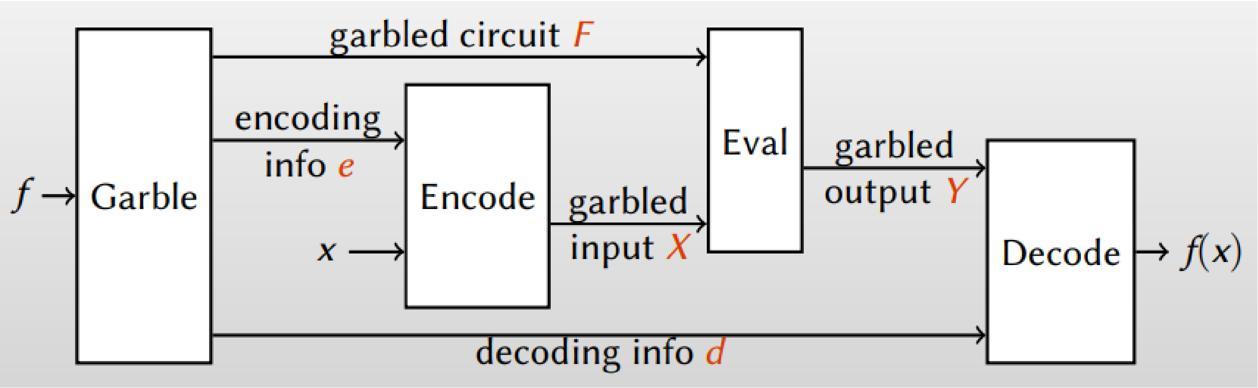


- Homomorphic Encryption
  - Perform computations on its encrypted data without first decrypting it.



Outline

- ~~What is Secure Computation?~~
- How does it work?
  - Yao's Protocol (Garbled Circuit)
  - Homomorphic Encryption (Next Lecture)



Yao's Millionaires’ Problem

$x > y$?

- Input domain: $x, y \in \{1,2\}$
  - Alice’s input: x =2
  - Bob’s input: $y = 3$

Yao's Millionaires' Problem

$x > y$?

- Input domain: $x, y \in \{1,2,3\}$
  - Alice's input: $x = 2$
  - Bob's input: $y = 3$
- Strawman solution:
  - Alice does the following :
    - Write truth table of function $\mathrm{f}(x, y) = x > y$?
    - For each possible input,
      choose random cryptographic key
    - Encrypt each output with corresponding keys
    - Randomly permute ciphertexts, send to Bob

x	y	$f(x, y)$

Yao's Millionaires' Problem

x > y ?

- Input domain: $x, y \in \{1,2\}$
  - Alice's input: $x = 2$
  - Bob's input: $y = 3$
- Strawman solution:
  - Alice does the following :
    - Write truth table of function $\mathrm{f}(x, y) = x > y?$
    - For each possible input,
      choose random cryptographic key
    - Encrypt each output with corresponding keys
    - Randomly permute ciphertexts, send to Bob

Somehow Bob obtains only "correct" encrypted keys: A_x, B_y

x	y	$f(x,y)$
A_1	B_1	$Enc_{A_1,B_1}(f(1,1))$
A_2	B_3	$Enc_{A_2,B_3}(f(2,3))$
A_1	B_3	$Enc_{A_1,B_3}(f(1,3))$
A_2	B_1	$Enc_{A_2,B_1}(f(2,1))$
A_2	B_2	$Enc_{A_2,B_2}(f(2,2))$
A_1	B_2	$Enc_{A_1,B_2}(f(1,2))$
A_3	B_1	$Enc_{A_3,B_1}(f(3,1))$
A_3	B_2	$Enc_{A_3,B_2}(f(3,2))$
A_3	B_3	$Enc_{A_3,B_3}(f(3,3))$

Yao's Millionaires’ Problem

x > y ?

- Input domain: $x, y \in \{1,2\}$
  - Alice’s input: $x = 2$
  - Bob’s input: $y = 3$
- Strawman solution:
  - Alice does the following :
    - Write truth table of function $\mathrm{f}(x, y) = x > y$?
    - For each possible input,
      choose random cryptographic key
    - Encrypt each output with corresponding keys
    - Randomly permute ciphertexts, send to Bob

  Somehow Bob obtains only “correct” encrypted keys: A_x, B_y
  - Bob learns only f(x, y)

x	y	$f(x,y)$
A_1	B_1	$Enc_{A_1,B_1}(f(1,1))$
A_2	B_3	$Enc_{A_2,B_3}(f(2,3))$
A_1	B_3	$Enc_{A_1,B_3}(f(1,3))$
A_2	B_1	$Enc_{A_2,B_1}(f(2,1))$
A_2	B_2	$Enc_{A_2,B_2}(f(2,2))$
A_1	B_2	$Enc_{A_1,B_2}(f(1,2))$
A_3	B_1	$Enc_{A_3,B_1}(f(3,1))$
A_3	B_2	$Enc_{A_3,B_2}(f(3,2))$
A_3	B_3	$Enc_{A_3,B_3}(f(3,3))$

Yao's Millionaires' Problem

x > y ?

- Input domain: $x, y \in \{1,2\}$
  - Alice's input: $x = 1$
  - Bob's input: $y = 2$
- Strawman solution
- Problem:
  - How does Bob learn A_x, B_y?
    - Using Oblivious Transfer (discuss later)
  - And, cost scales with the truth table size of f
    - Idea: instead of encrypting outputs, we encrypt each gate of circuit f

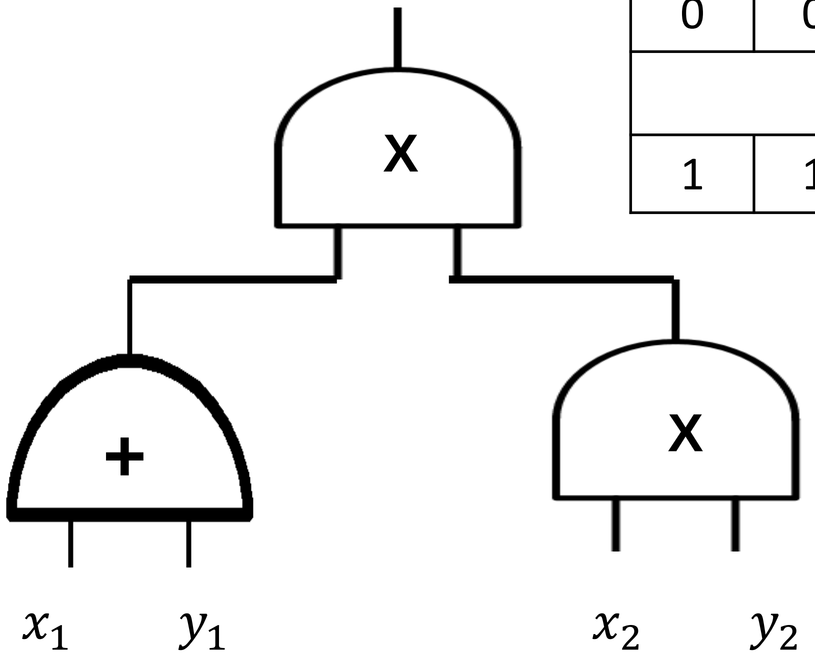
Somehow Bob obtains only "correct" encrypted keys: A_x, B_y

x	y	$f(x,y)$
A_1	B_1	$Enc_{A_1,B_1}(f(1,1))$
A_2	B_3	$Enc_{A_2,B_3}(f(2,3))$
A_1	B_3	$Enc_{A_1,B_3}(f(1,3))$
A_2	B_1	$Enc_{A_2,B_1}(f(2,1))$
A_2	B_2	$Enc_{A_2,B_2}(f(2,2))$
A_1	B_2	$Enc_{A_1,B_2}(f(1,2))$
A_3	B_1	$Enc_{A_3,B_1}(f(3,1))$
A_3	B_2	$Enc_{A_3,B_2}(f(3,2))$
A_3	B_3	$Enc_{A_3,B_3}(f(3,3))$

Yao's Protocol

- Instead of encrypting outputs f, we encrypt each gate of the circuit f
- For example, $f(x_1, x_2, y_1, y_2) = (x_1 + y_1)x_2y_2$
  - Where Alice's input is x_1, x_2 and Bob's input is y_1, y_2

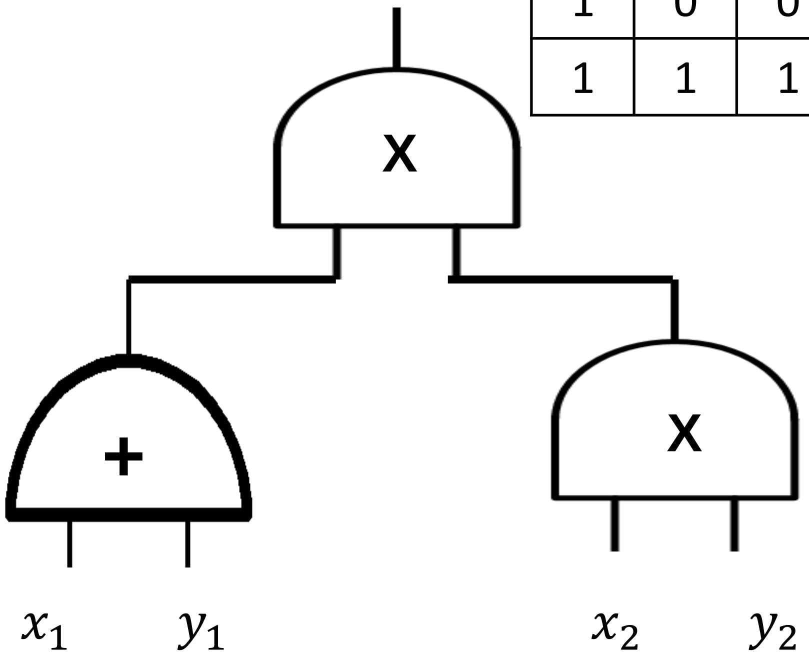
x_1	x_2	y_1	y_2	$f()$
0	0	0	0	0
…				
1	1	1	1	1



Yao's Protocol

- Instead of encrypting outputs f, we encrypt each gate of the circuit f
- For example, $f(x_1, x_2, y_1, y_2) = (x_1 + y_1)x_2y_2$
  - Where Alice's input is x_1, x_2 and Bob's input is y_1, y_2

$?$	$?$	X
0	0	0
0	1	0
1	0	0
1	1	1

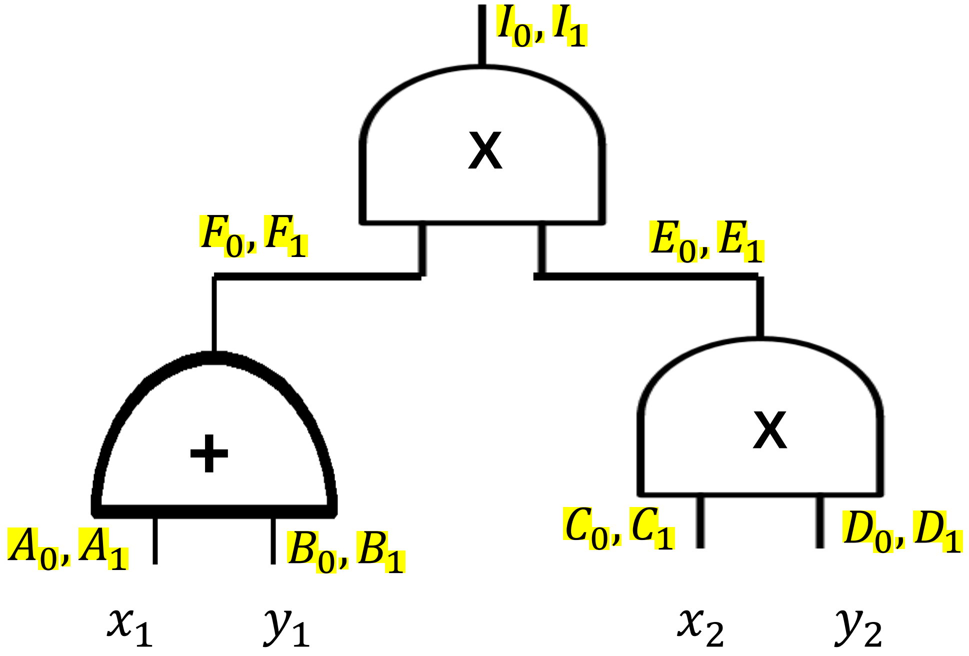


x_1	y_1	$+$
0	0	0
0	1	1
1	0	1
1	1	0

x_2	y_2	X
0	0	0
0	1	0
1	0	0
1	1	1

Yao's Protocol

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire

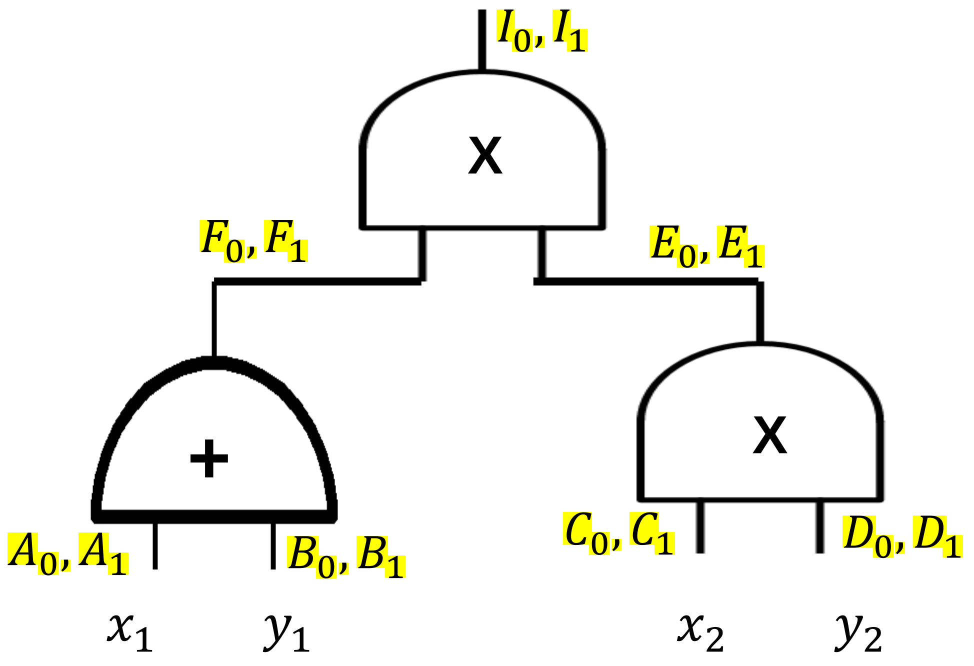


Yao’s Protocol

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire
    - “Encrypt” truth table of each gate

$Enc_{A_0,B_0}(F_0)$
$Enc_{A_0,B_1}(F_1)$
$Enc_{A_1,B_0}(F_1)$
$Enc_{A_1,B_1}(F_0)$

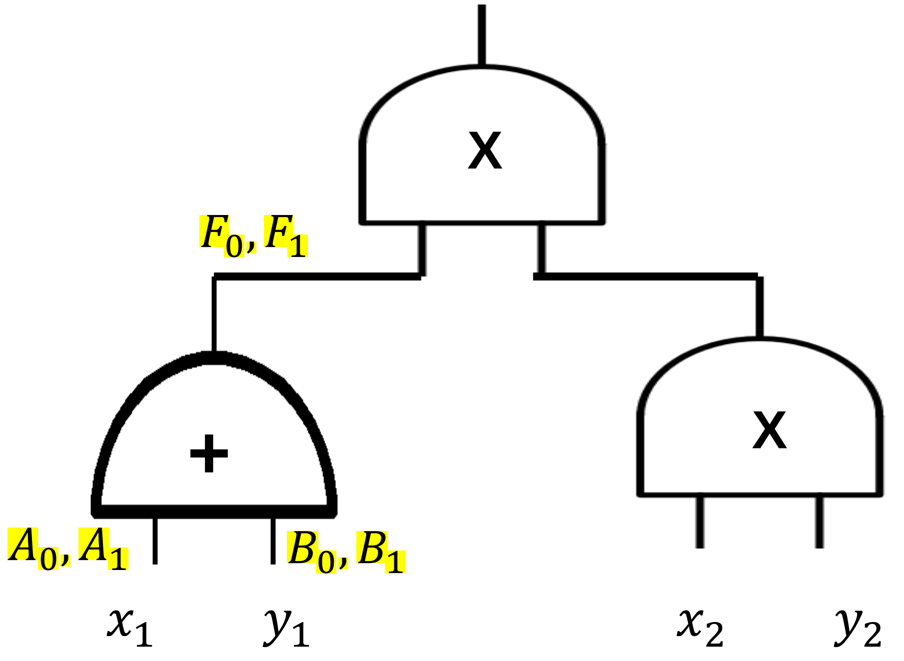
x_1	y_1	$+$
0	0	0
0	1	1
1	0	1
1	1	0



Yao's Protocol

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire
    - "Encrypt" truth table of each gate

$Enc_{A_0,B_0}(F_0)$
$Enc_{A_0,B_1}(F_1)$
$Enc_{A_1,B_0}(F_1)$
$Enc_{A_1,B_1}(F_0)$

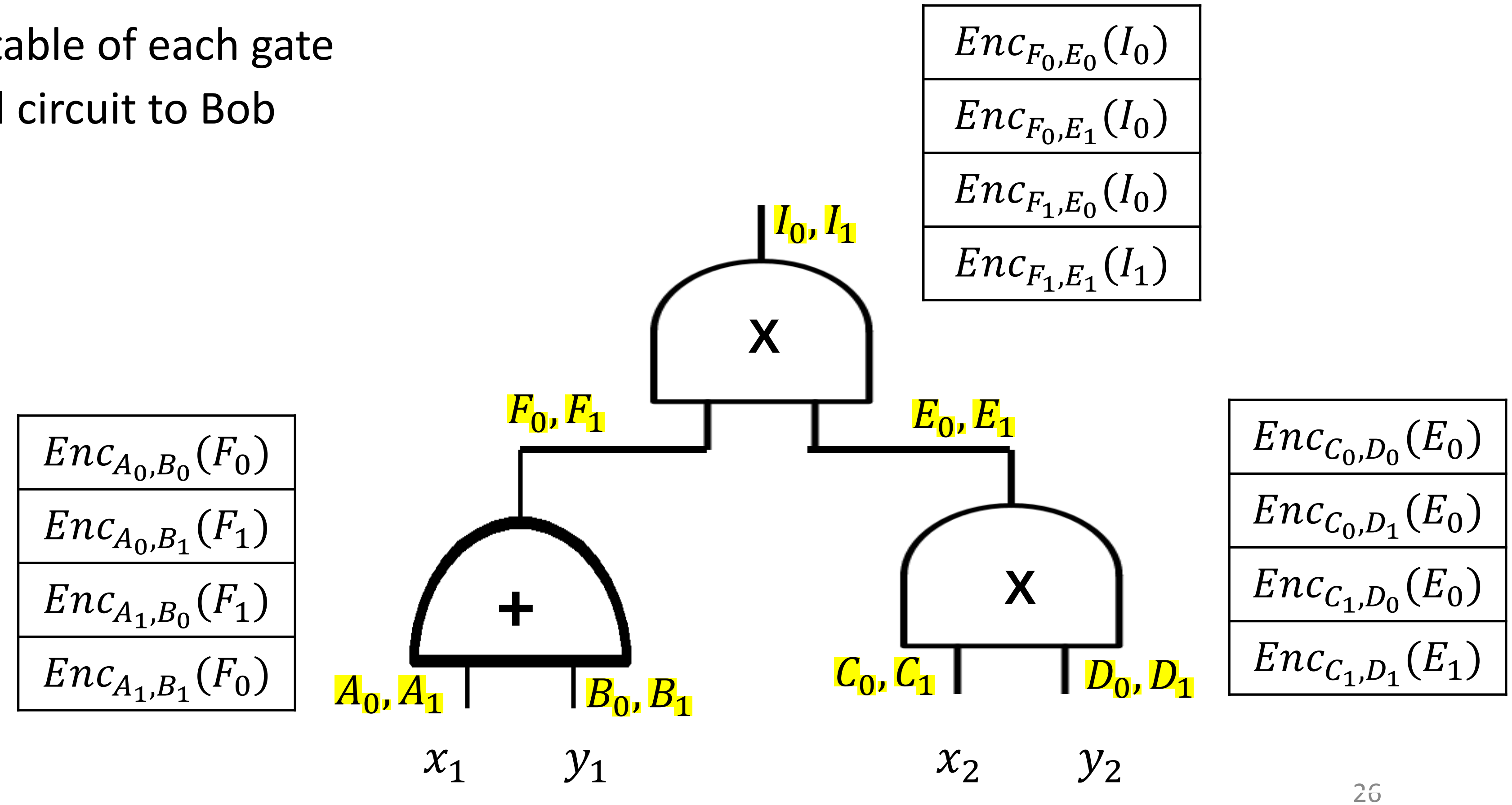


Yao's Protocol

Garbled circuit ≡ all encrypted gates

Garbled encoding ≡ one label per wire

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire
    - "Encrypt" truth table of each gate
    - Send the garbled circuit to Bob

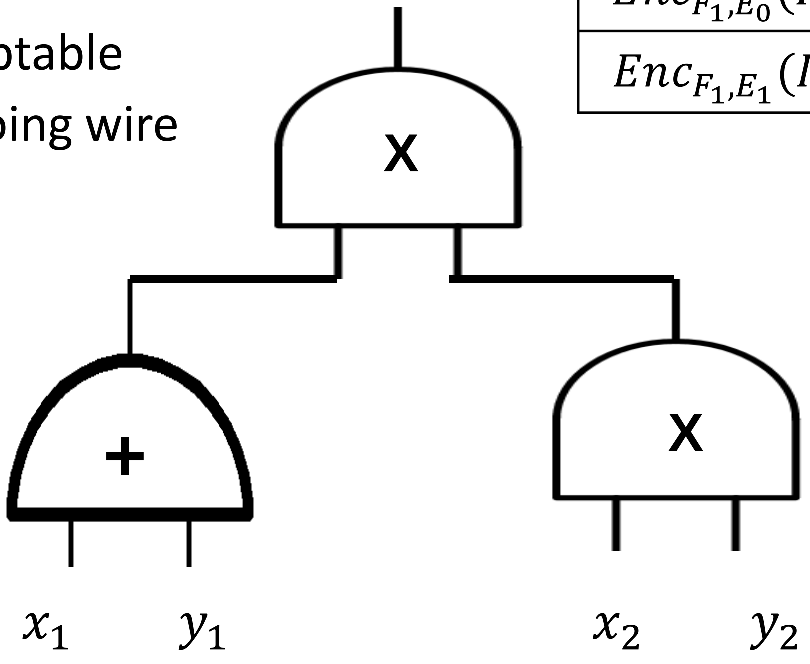


Yao's Protocol

Somehow Bob obtains only "correct" encrypted keys

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire
    - "Encrypt" truth table of each gate
    - Send the garbled circuit to Bob
  - (Bob) Garbled evaluation:
    - Only one ciphertext per gate is decryptable
    - Result of decryption is value on outgoing wire

$Enc_{F_0,E_0}(I_0)$
$Enc_{F_0,E_1}(I_0)$
$Enc_{F_1,E_0}(I_0)$
$Enc_{F_1,E_1}(I_1)$



$Enc_{A_0,B_0}(F_0)$
$Enc_{A_0,B_1}(F_1)$
$Enc_{A_1,B_0}(F_1)$
$Enc_{A_1,B_1}(F_0)$

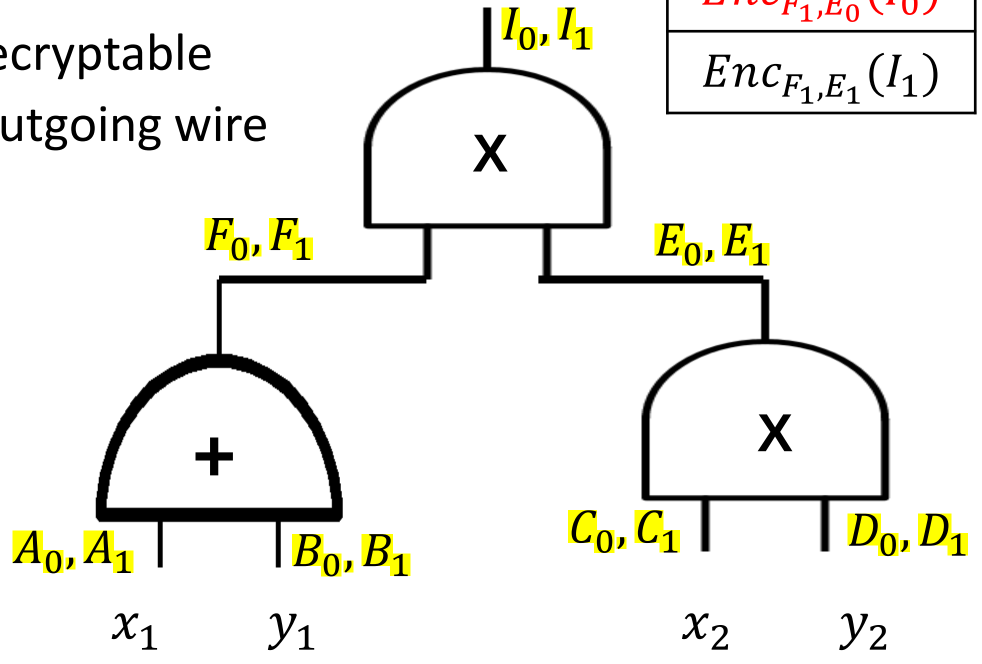
$Enc_{C_0,D_0}(E_0)$
$Enc_{C_0,D_1}(E_0)$
$Enc_{C_1,D_0}(E_0)$
$Enc_{C_1,D_1}(E_1)$

Yao's Protocol

- Construction:
  - (Alice) Garbling a circuit:
    - Pick random labels on each wire
    - "Encrypt" truth table of each gate
    - Send the garbled circuit to Bob
  - (Bob) Garbled evaluation:
    - Only one ciphertext per gate is decryptable
    - Result of decryption is value on outgoing wire

Somehow Bob obtains only "correct" encrypted keys

$Enc_{F_0,E_0}(I_0)$
$Enc_{F_0,E_1}(I_0)$
$Enc_{F_1,E_0}(I_0)$
$Enc_{F_1,E_1}(I_1)$



$Enc_{A_0,B_0}(F_0)$
$Enc_{A_0,B_1}(F_1)$
$Enc_{A_1,B_0}(F_1)$
$Enc_{A_1,B_1}(F_0)$

$Enc_{C_0,D_0}(E_0)$
$Enc_{C_0,D_1}(E_0)$
$Enc_{C_1,D_0}(E_0)$
$Enc_{C_1,D_1}(E_1)$

Sample quiz:

- What is MPC?

Sample quiz:

- What is the purpose of "garbling" in Garbled Circuit?