601.445/601.645
Practical Cryptographic Systems

MPC and Private Computation Continued

Instructor: Alishah Chator

Housekeeping

- Assignment 2 due Tuesday 10/26 at 11:59pm
- Weekly HW#3 Due Thursday 10/28 at 11:59pm
- Midterm 11/1
    - In Class, Let us know ASAP if you cannot attend that day

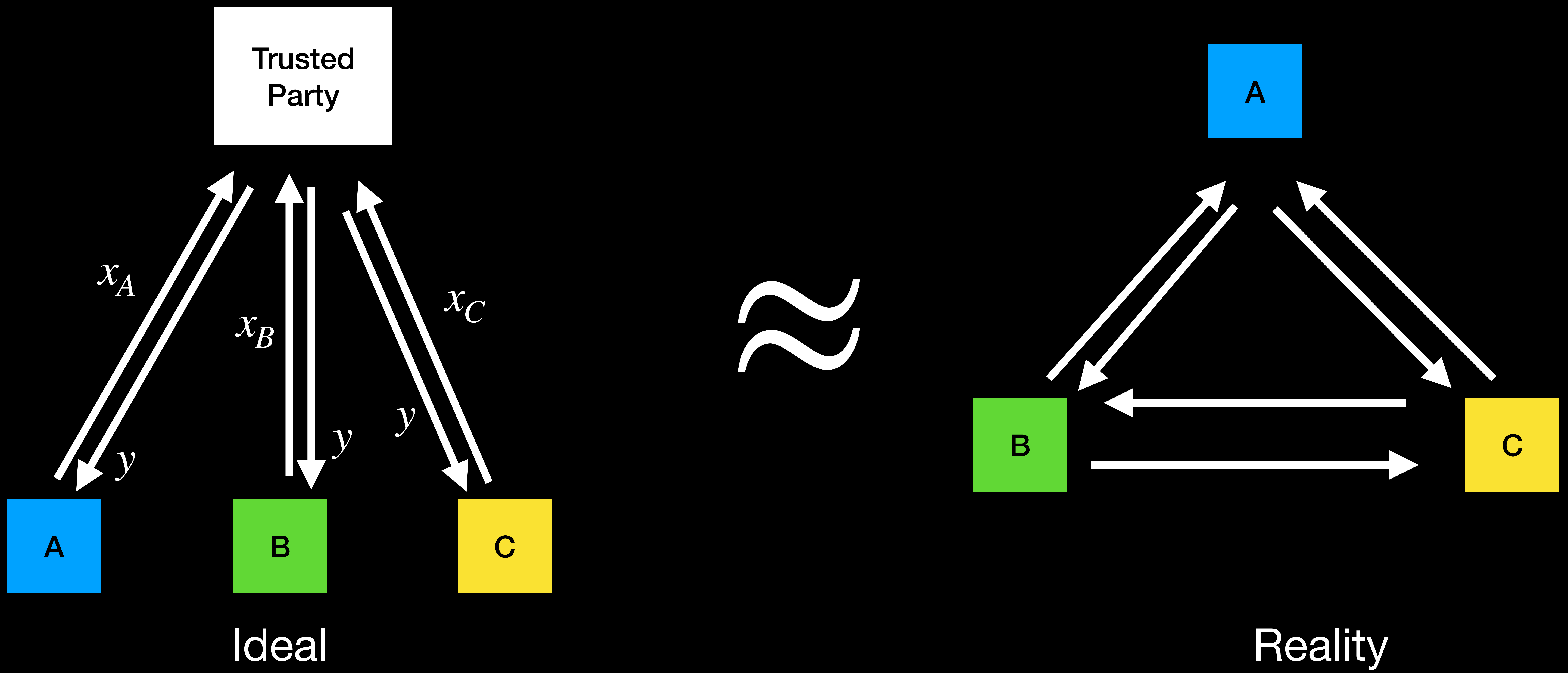
News

Last Time

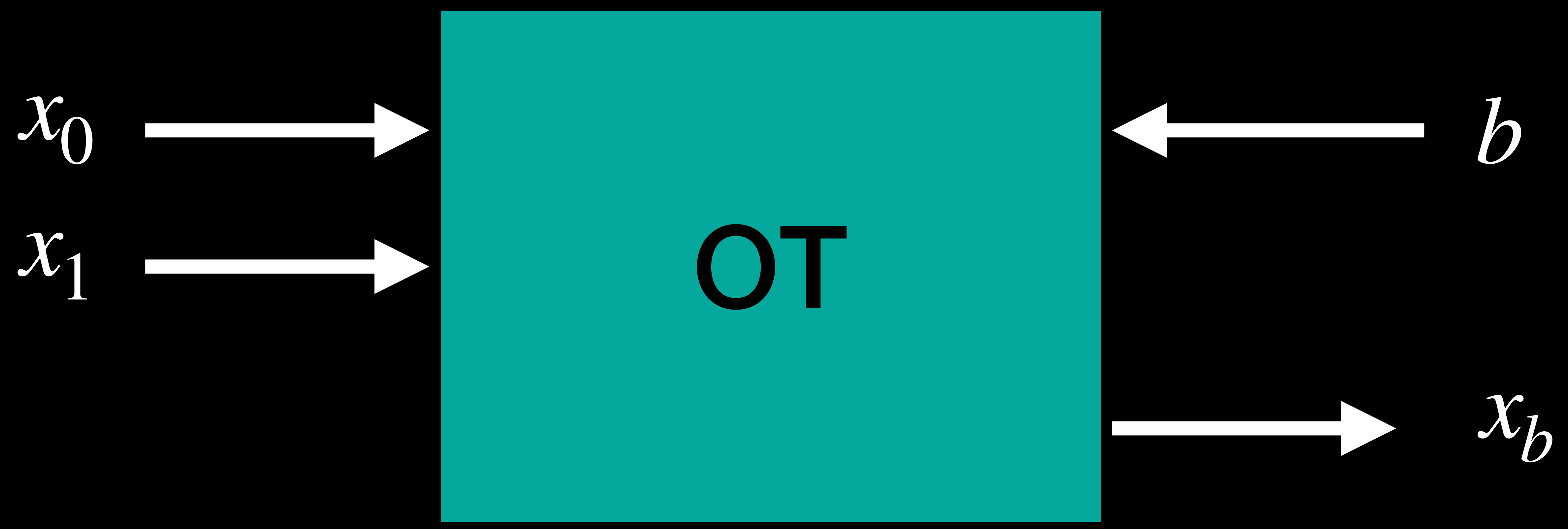
MPC Formally

- A Multiparty Computation Protocol involves a set of parties $\{A, B, C, \ldots\}$, a private input for each party $\{x_A, x_B, x_C, \ldots\}$ and some functionality to jointly compute $f(x_A, x_B, x_C, \ldots) = y$
  - Each party should learn nothing besides the output y
  - Crucially, that means the input of all other parties should be hidden

MPC: Ideal vs Reality



1-out-of-2 Oblivious Transfer



Private Information Retrieval (PIR)

- Goal: Query a server's database without:
  - The server learning which entry was looked up
  - The client learning other entries of the database
- How might we do this naively?
- Can we use OT to build this?

Today

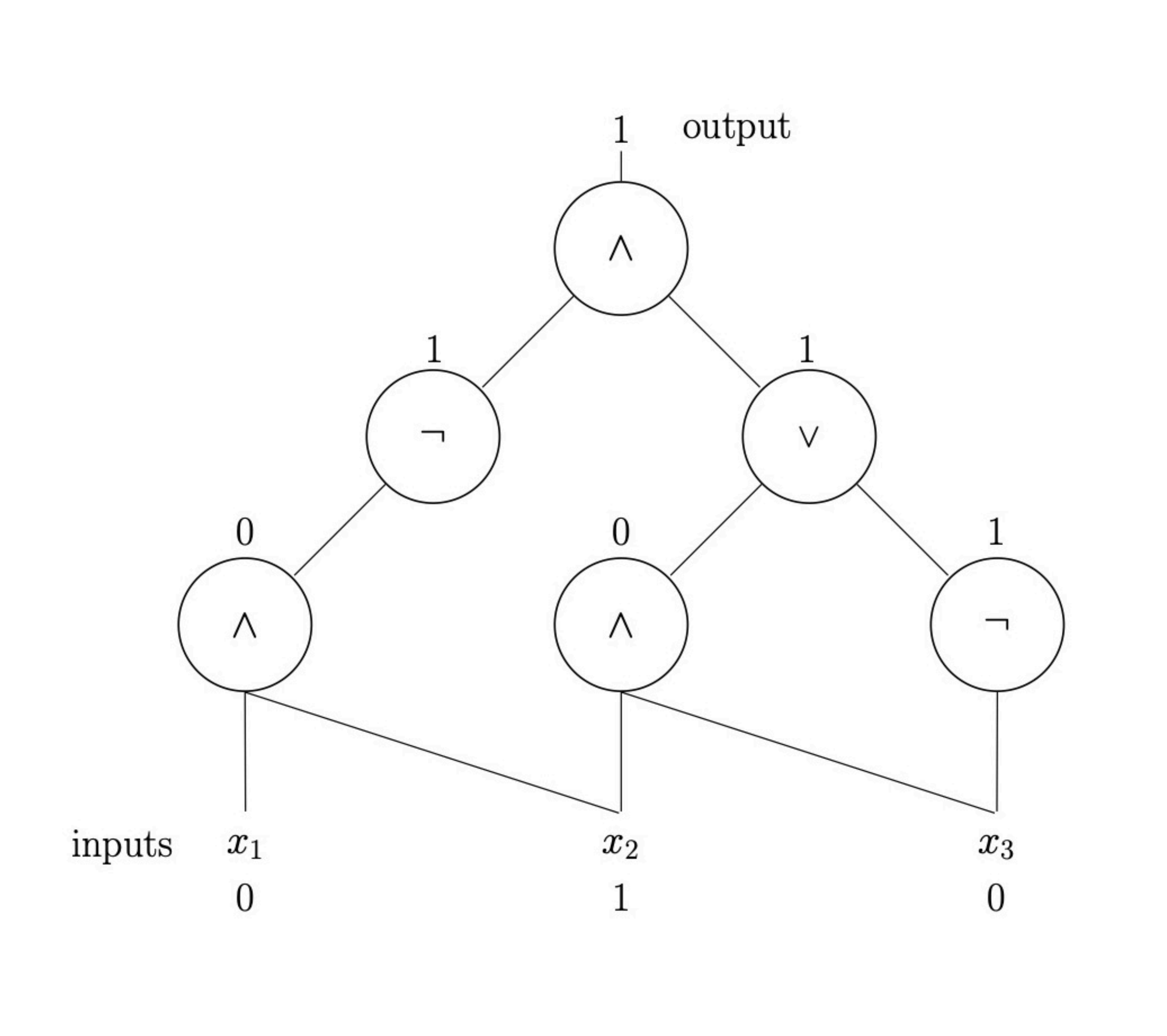
- How can we build general MPC for two parties
- How can we build MPC for an arbitrary number of parties

Boolean Circuits

- Way of breaking down functions into logical operations (AND, OR, NOT)

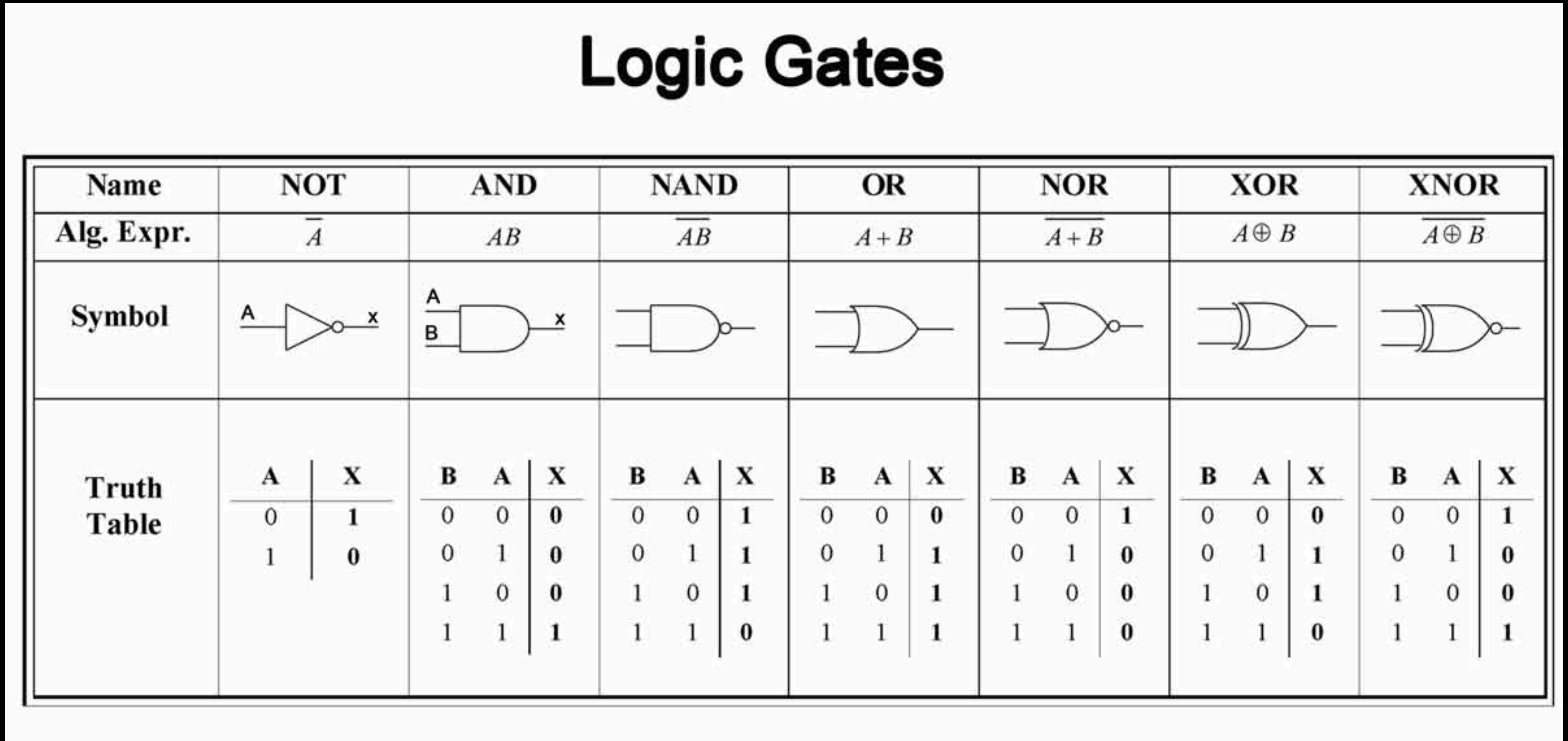
Boolean Circuits

- Way of breaking down functions into logical operations (AND, OR, NOT)



Gates and Truth Tables

- The basic gates in a boolean circuit are AND, OR, NOT, XOR
- Their operations are described in truth tables



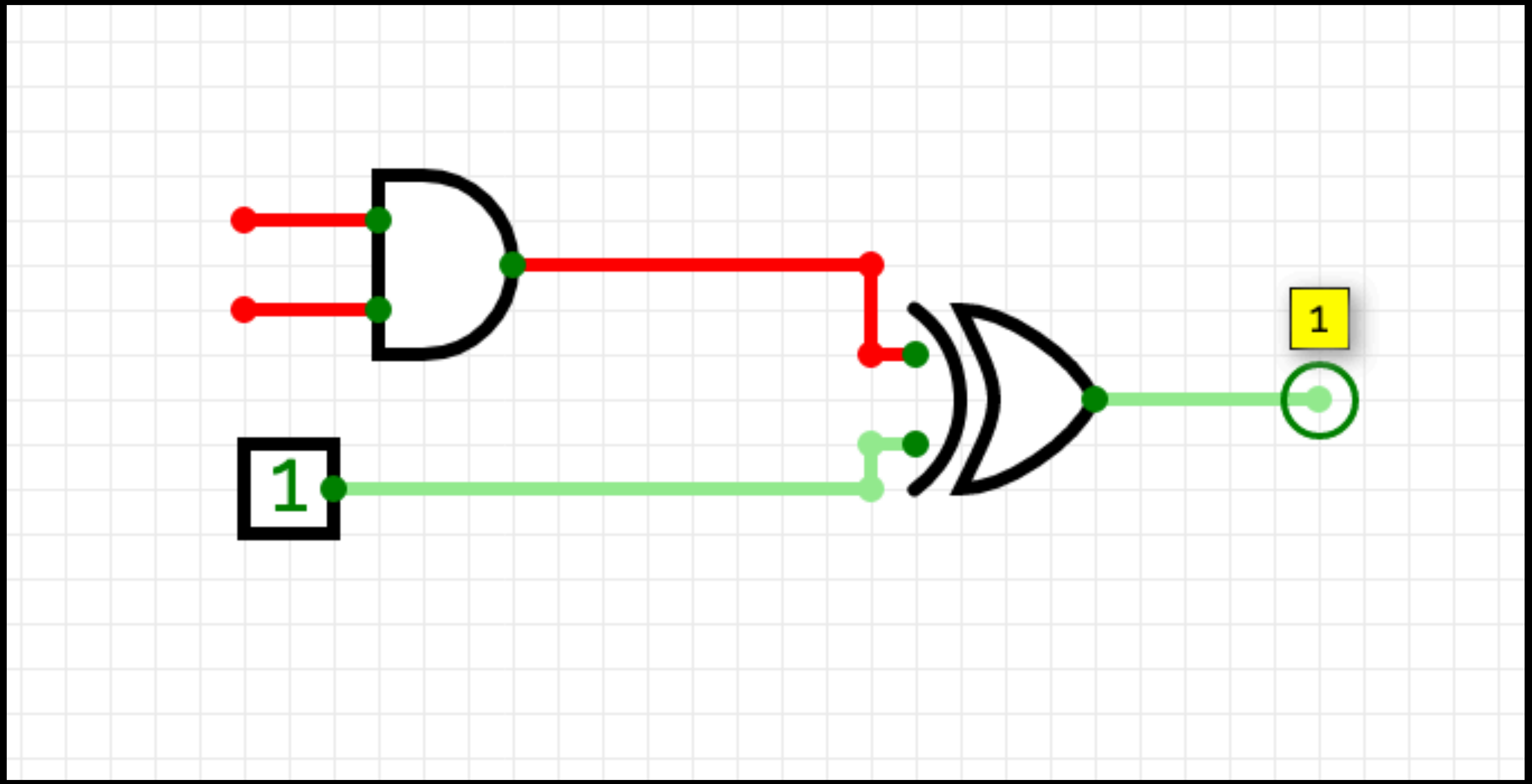
Logic Gates

Name	NOT	AND	NAND	OR	NOR	XOR	XNOR
Alg. Expr.	$\overline{A}$	AB	$\overline{AB}$	$A+B$	$\overline{A+B}$	$A \oplus B$	$\overline{A \oplus B}$
Symbol	A x	A B x					
Truth Table	A X 0 1 1 0	B A X 0 0 0 0 1 0 1 0 0 1 1 1	B A X 0 0 1 0 1 1 1 0 1 1 1 0	B A X 0 0 0 0 1 1 1 0 1 1 1 1	B A X 0 0 1 0 1 0 1 0 0 1 1 0	B A X 0 0 0 0 1 1 1 0 1 1 1 0	B A X 0 0 1 0 1 0 1 0 0 1 1 1

Universality

- Some combinations of gates are **Universal**
    - These gates can be combined to build any function
- Examples are {NAND}, {AND,OR,NOT}
- Often in cryptography we are limited to {AND, XOR}

NAND from AND + XOR



Two-Party Secure Computation from circuits?

- We can build any function from {AND, XOR}, how does this help us?

Two-Party Secure Computation from circuits?

- We can build any function from {AND, XOR}, how does this help us?

- IDEA: Build a circuit that can only be run on one input!

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
    - The garbler constructs a boolean circuit for the desired function

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their input to the evaluator

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - The evaluator evaluates the circuit

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - **The garbler constructs a boolean circuit for the desired function**
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - The evaluator evaluates the circuit

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
    - **The garbler constructs a boolean circuit for the desired function**
        - **Universality of NAND (can be built from {AND,XOR})**
    - The garbler garbles the circuit
    - The garbler sends the garbled circuit and their encrypted input to the evaluator
    - The evaluator evaluates the circuit

Garbled Circuits

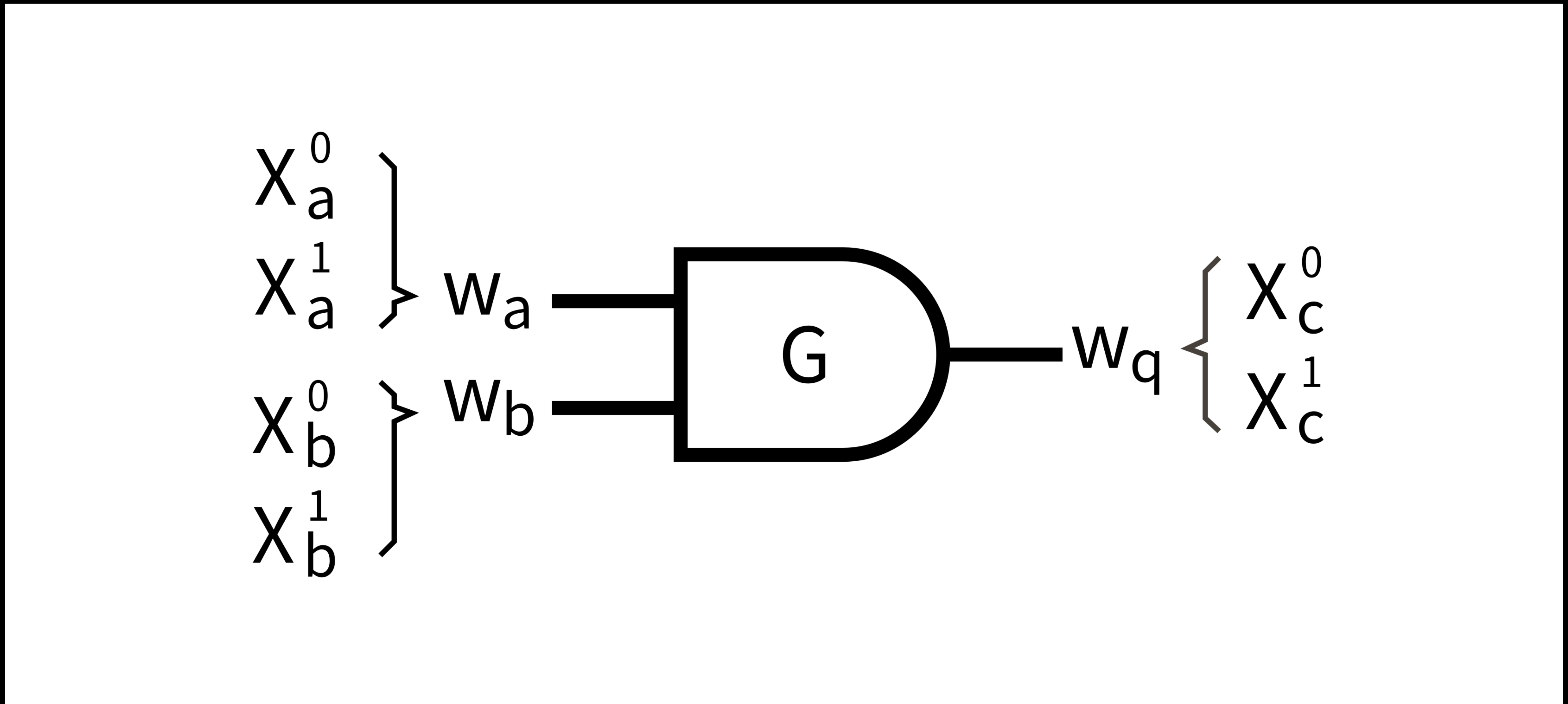
- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - **The garbler garbles the circuit**
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - The evaluator evaluates the circuit

Garbling

- Garbling is the process of encrypting a truth table, so the corresponding gate's result is hidden

Garbling

- Garbling is the process of encrypting a truth table, so the corresponding gate's result is hidden



b \ a	**0**	**1**
0	0	0
1	0	1

$\rightarrow$

b \ a	X_a^0	X_a^1
X_b^0	X_c^0	X_c^0
X_b^1	X_c^0	X_c^1

$\rightarrow$

b \ a	X_a^0	X_a^1
X_b^0	$E_{X_a^0, X_b^0}(X_c^0)$	$E_{X_a^1, X_b^0}(X_c^0)$
X_b^1	$E_{X_a^0, X_b^1}(X_c^0)$	$E_{X_a^1, X_b^1}(X_c^1)$

Garbling

- This process is done for every gate in the circuit
- If you only know one label for each input to a gate you can only evaluate the gate one way
  - You only get one wire label as output from each gate

Garbling

- This process is done for every gate in the circuit
- If you only know one label for each input to a gate you can only evaluate the gate one way
  - You only get one wire label as output from each gate
- Does order matter for the garbled truth table?

Garbling

- This process is done for every gate in the circuit
- If you only know one label for each input to a gate you can only evaluate the gate one way
  - You only get one wire label as output from each gate
- Does order matter for the garbled truth table?
- What about the inputs wires to the circuit?

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - **The garbler sends the garbled circuit and their encrypted input to the evaluator**
  - The evaluator evaluates the circuit

Sending the Garbled Circuit

- The garbler sends over the garbled truth tables for each gate
- The garbler also sends over the wire labels corresponding to their input
    - Why don’t these reveal the garblers input?
- Does the Evaluator have everything they need to evaluate?

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - **The evaluator gets their input labels from the garbler**
  - The evaluator evaluates the circuit

Getting the Evaluator’s labels

- Cannot directly ask the garbler for the labels without revealing evaluator’s input

Getting the Evaluator’s labels

- Cannot directly ask the garbler for the labels without revealing evaluator’s input
- Two possible labels for each input wire, need to obtain exactly 1 label for each. Do we have a primitive for this?

Getting the Evaluator’s labels

- Cannot directly ask the garbler for the labels without revealing evaluator’s input
- Two possible labels for each input wire, need to obtain exactly 1 label for each. Do we have a primitive for this?
  - Use 1-out-of-2 Oblivious Transfer!

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - The evaluator gets their input labels from the garbler
  - **The evaluator evaluates the circuit**

Garbled Circuits

- A **Garbled Circuit** is a two party secure computation protocol between a Garbler and an Evaluator.
- Steps:
  - The garbler constructs a boolean circuit for the desired function
  - The garbler garbles the circuit
  - The garbler sends the garbled circuit and their encrypted input to the evaluator
  - The evaluator gets their input labels from the garbler
  - **The evaluator evaluates the circuit**
    - **Uses the given wire labels to evaluate the garbled truth tables one by one**

Output

- Eventually the evaluator arrives at an output label
- They will need to communicate with garbler to obtain the corresponding output

Output

- Eventually the evaluator arrives at an output label
- They will need to communicate with garbler to obtain the corresponding output
  - The garbler can send the mapping over
  - The evaluator can tell the garbler the resulting label

Output

- Eventually the evaluator arrives at an output label
- They will need to communicate with garbler to obtain the corresponding output
  - The garbler can send the mapping over
  - The evaluator can tell the garbler the resulting label
- In either case, we see we only have **semi-honest security**

Output

- Eventually the evaluator arrives at an output label
- They will need to communicate with garbler to obtain the corresponding output
  - The garbler can send the mapping over
  - The evaluator can tell the garbler the resulting label
- In either case, we see we only have **semi-honest security**
- Should be easy to see that inputs remain hidden for both parties

Limitations

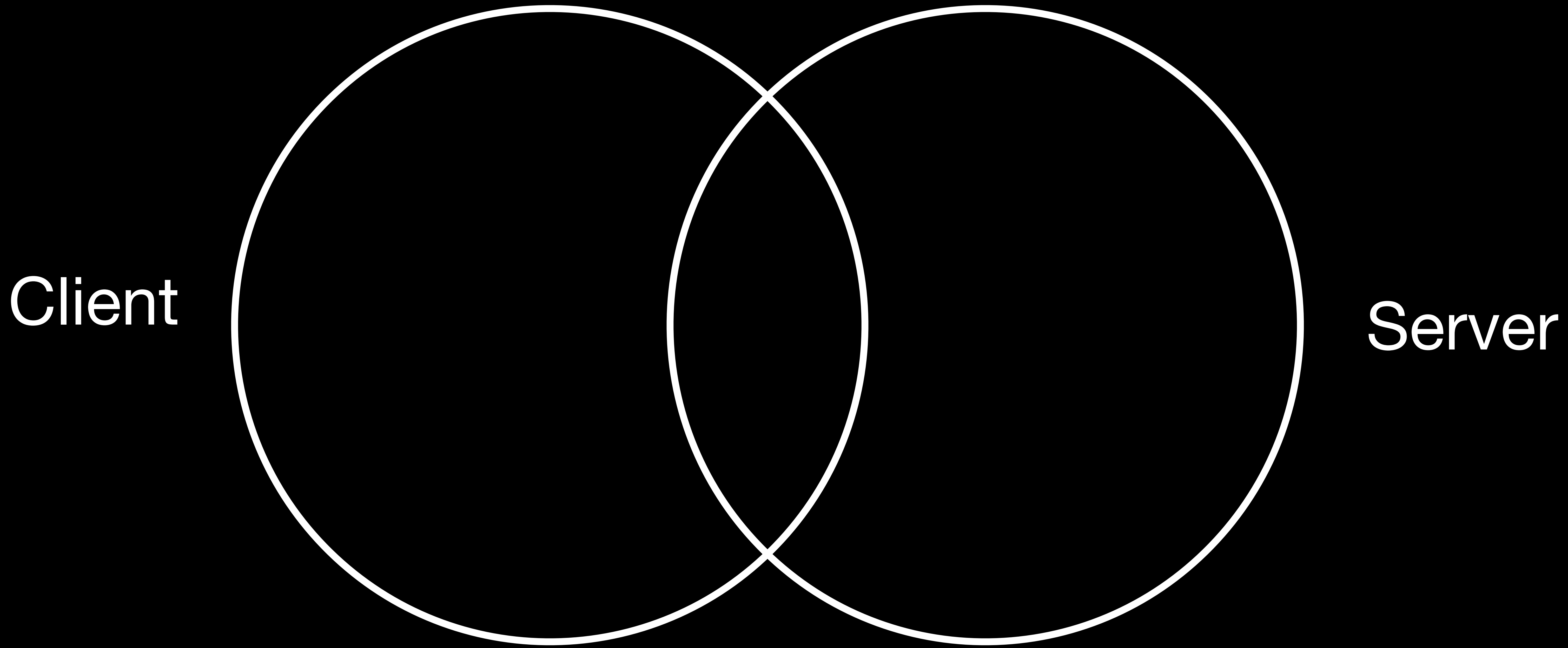
- Only support two parties
- Computation and Communication cost depend on the number of gates
  - Not all functions have concise circuit representations

So what can we do with garbled circuits

- Ideal for the secure evaluations of functions with low depth/number of gates
- PRFs (i.e. AES) have workable circuit sizes
- Why might we want to securely evaluate a PRF?

Private Set Intersection (PSI)

- Functionality: Find the overlapping elements between two party’s inputs



Building PSI

- Attempt 1: Server sends its database to the client

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: ???

Oblivious PRF

- An **Oblivious PRF** is a two party protocol where one party inputs a key k and a second party inputs a value x to jointly compute the PRF output $f_k(x)$

Oblivious PRF

- An **Oblivious PRF** is a two party protocol where one party inputs a key k and a second party inputs a value x to jointly compute the PRF output $f_k(x)$
- How can we build it?

Oblivious PRF

- An **Oblivious PRF** is a two party protocol where one party inputs a key k and a second party inputs a value x to jointly compute the PRF output $f_k(x)$
- How can we build it?
    - Construct a Garbled Circuit for f where the parties inputs are k and x respectively

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: Garbled Circuits

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: Garbled Circuits
  - Server samples a key k, and builds a Garbled Circuit C for f

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: Garbled Circuits
  - Server samples a key k, and builds a Garbled Circuit C for f
  - Server computes $f_k(y)$ for all $y \in$ database

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: Garbled Circuits
  - Server samples a key k, and builds a Garbled Circuit C for f
  - Server computes $f_k(y)$ for all $y \in$ database
  - Server sends $C, f_k(y)\ldots$ to client

Building PSI

- Attempt 1: Server sends its database to the client
- Attempt 2: Server hashes its database and sends that to the client
- Attempt 3: Garbled Circuits
    - Server samples a key k, and builds a Garbled Circuit C for f
    - Server computes $f_k(y)$ for all $y \in$ database
    - Server sends $C, f_k(y)\ldots$ to client
    - Client evaluates C for each of its inputs x and compares the results to the hashed database

Applications of PSI

- Private contact discovery
- Client-side scanning
- Contact tracing