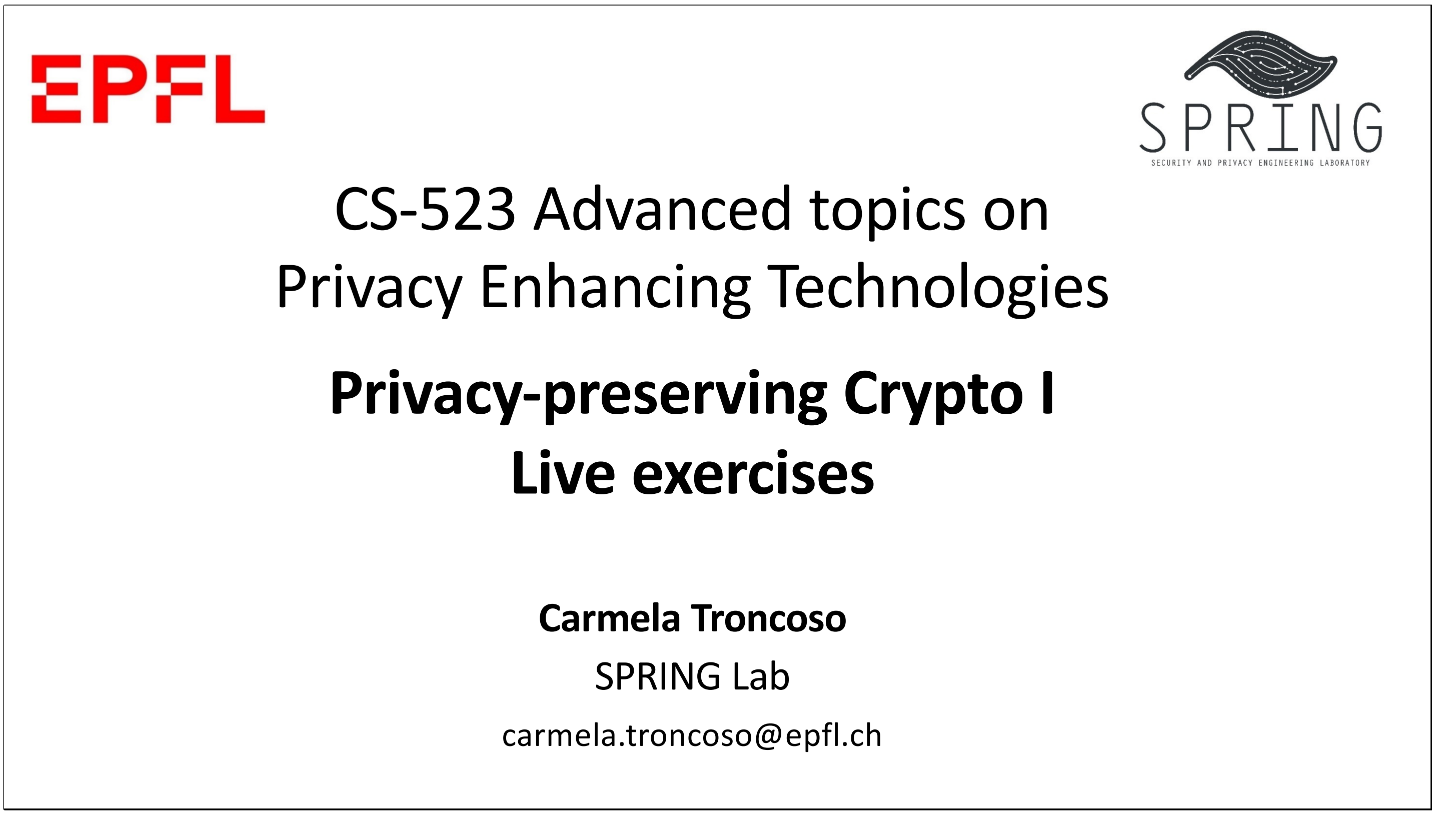


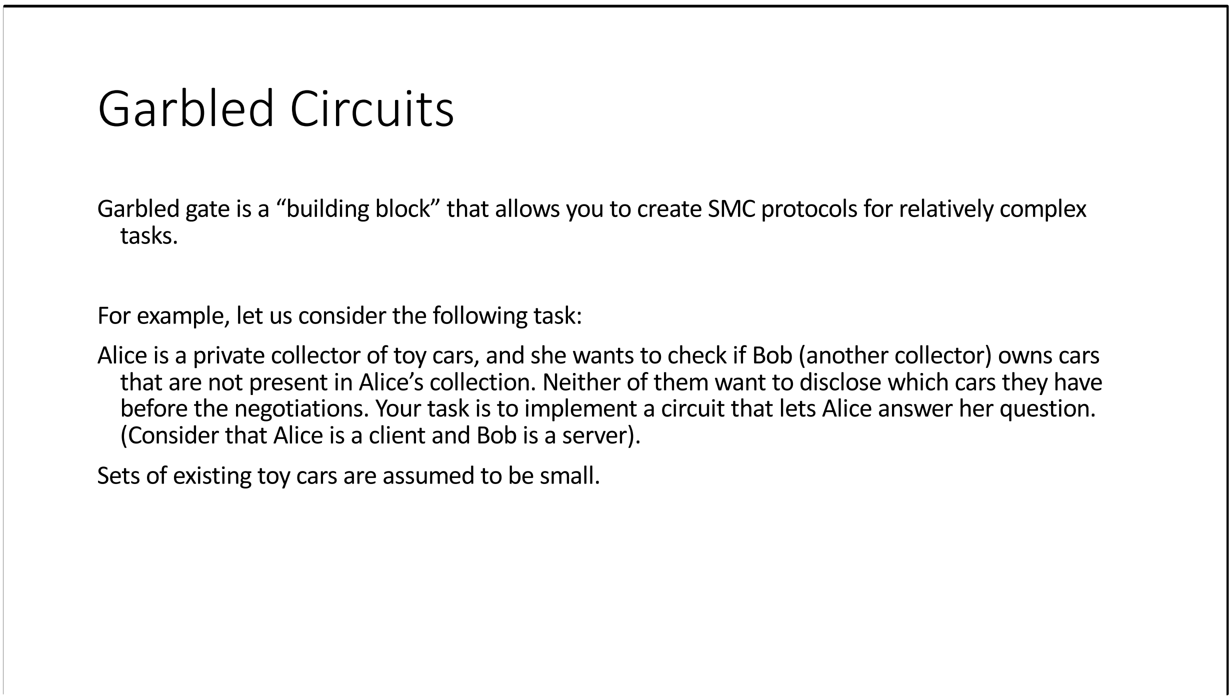
CS-523 Advanced topics on Privacy Enhancing Technologies

Privacy-preserving Crypto I
Live exercises

Carmela Troncoso

SPRING Lab

carmela.troncoso@epfl.ch



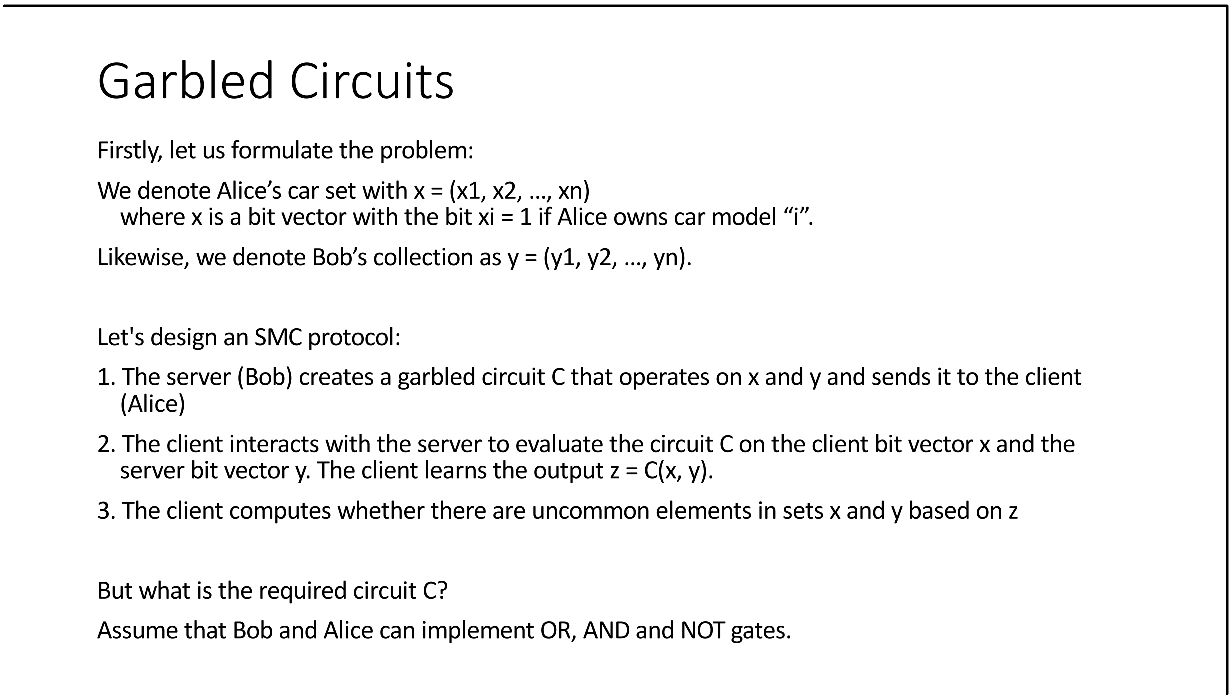
Garbled Circuits

Garbled gate is a “building block” that allows you to create SMC protocols for relatively complex tasks.

For example, let us consider the following task:

Alice is a private collector of toy cars, and she wants to check if Bob (another collector) owns cars that are not present in Alice’s collection. Neither of them want to disclose which cars they have before the negotiations. Your task is to implement a circuit that lets Alice answer her question. (Consider that Alice is a client and Bob is a server).

Sets of existing toy cars are assumed to be small.



Garbled Circuits

Firstly, let us formulate the problem:

We denote Alice’s car set with $x = (x1, x2, \ldots, xn)$
where x is a bit vector with the bit $xi = 1$ if Alice owns car model “i”.

Likewise, we denote Bob’s collection as $y = (y1, y2, \ldots, yn)$.

Let's design an SMC protocol:

1. The server (Bob) creates a garbled circuit C that operates on x and y and sends it to the client (Alice)
2. The client interacts with the server to evaluate the circuit C on the client bit vector x and the server bit vector y. The client learns the output $z = C(x, y)$.
3. The client computes whether there are uncommon elements in sets x and y based on z

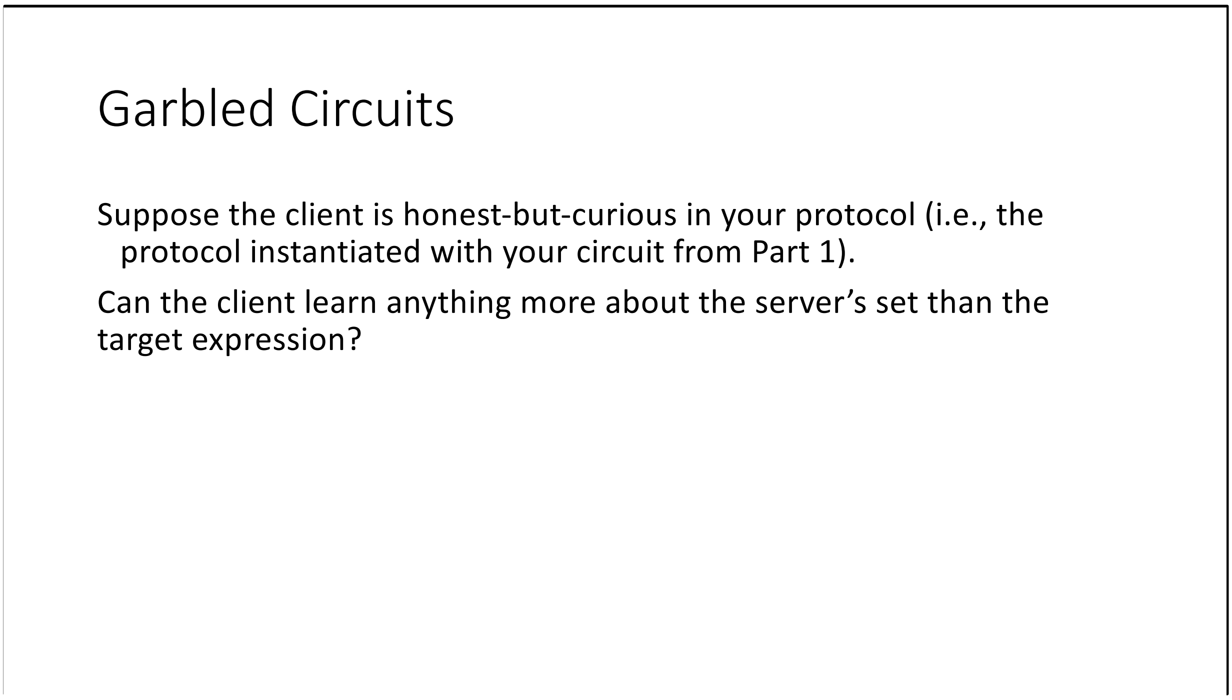
But what is the required circuit C?

Assume that Bob and Alice can implement OR, AND and NOT gates.

Let’s create an intermediate output t_i = NOT(x_i) AND y_i.

t_i is 1 iff x_i = 0 and y_i = 1, i.e., Bob has a car that Alice doesn’t. t_i is 0 otherwise
(we are not interested in cases where x_i = 1 and y_i = 0)

The final output is computed as c = t_1 OR t_2 OR ... OR t_n.
c is 1 if Bob owns cars that are not present in Alice’s collection, 0 otherwise



Garbled Circuits

Suppose the client is honest-but-curious in your protocol (i.e., the protocol instantiated with your circuit from Part 1).

Can the client learn anything more about the server's set than the target expression?

Without access to intermediate values nor inputs of the server, the honest but curious client cannot learn more. So the strict answer is yes, the 2-Party Garbled Circuit protocol is secure against honest-but-curious adversaries. Therefore the client cannot learn more than the specific output of the circuit/functionality.

In class we also discussed some other options that are beyond the strict question.
(1) What happens if the honest client is allowed to freely choose its input (but still run the protocol honestly)? Alice can for example find out if Bob has a specific car (exclude that car from the query string). One can argue that this leakage is allowed by the functionality hence not strictly a "leakage", yet it's important to discuss it.
(2) What happens if Alice can make multiple queries? She can repeat the above to extract Bob's set fully.

These last two questions show that even though the SMC building block is secure against honest adversaries, composing / repeating such protocols is not necessarily secure, and depends till where you extend the "honest" boundary.