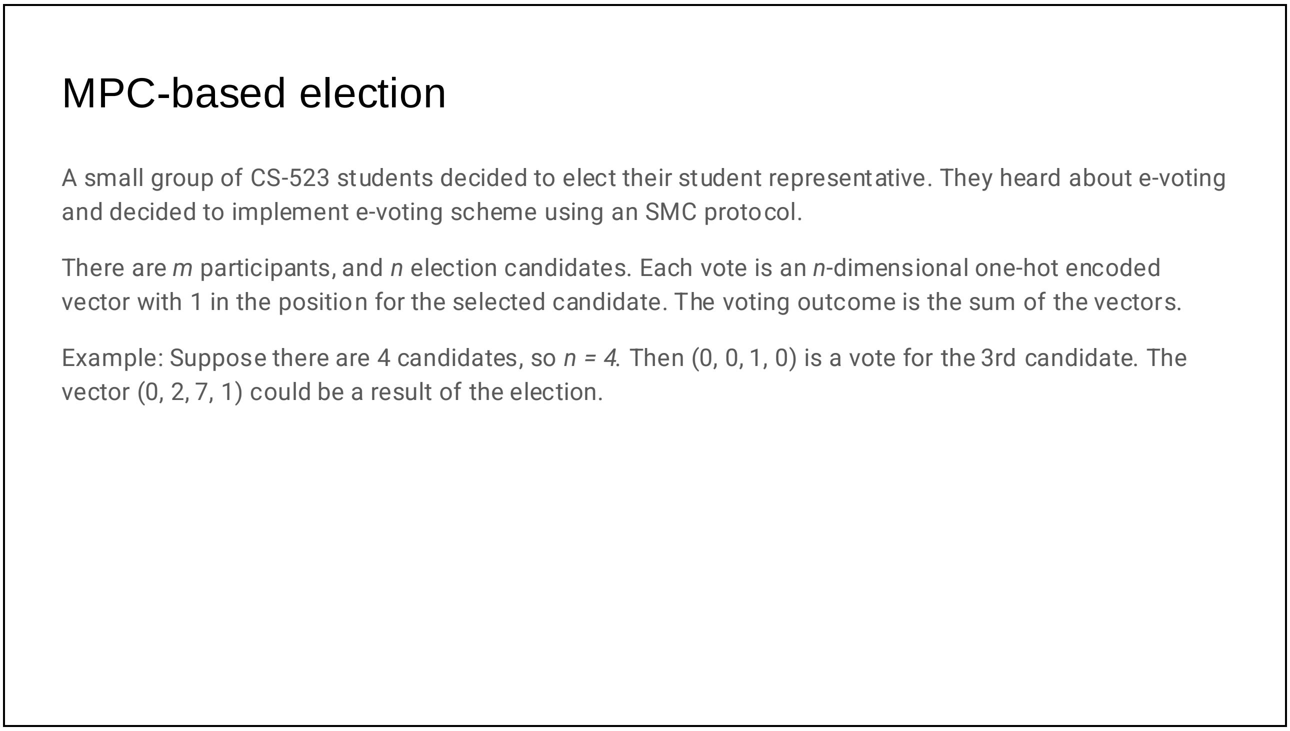
EPFL

SPRING

SECURITY AND PRIVACY ENGINEERING LABORATORY

Friday Live

Privacy-preserving Crypto I

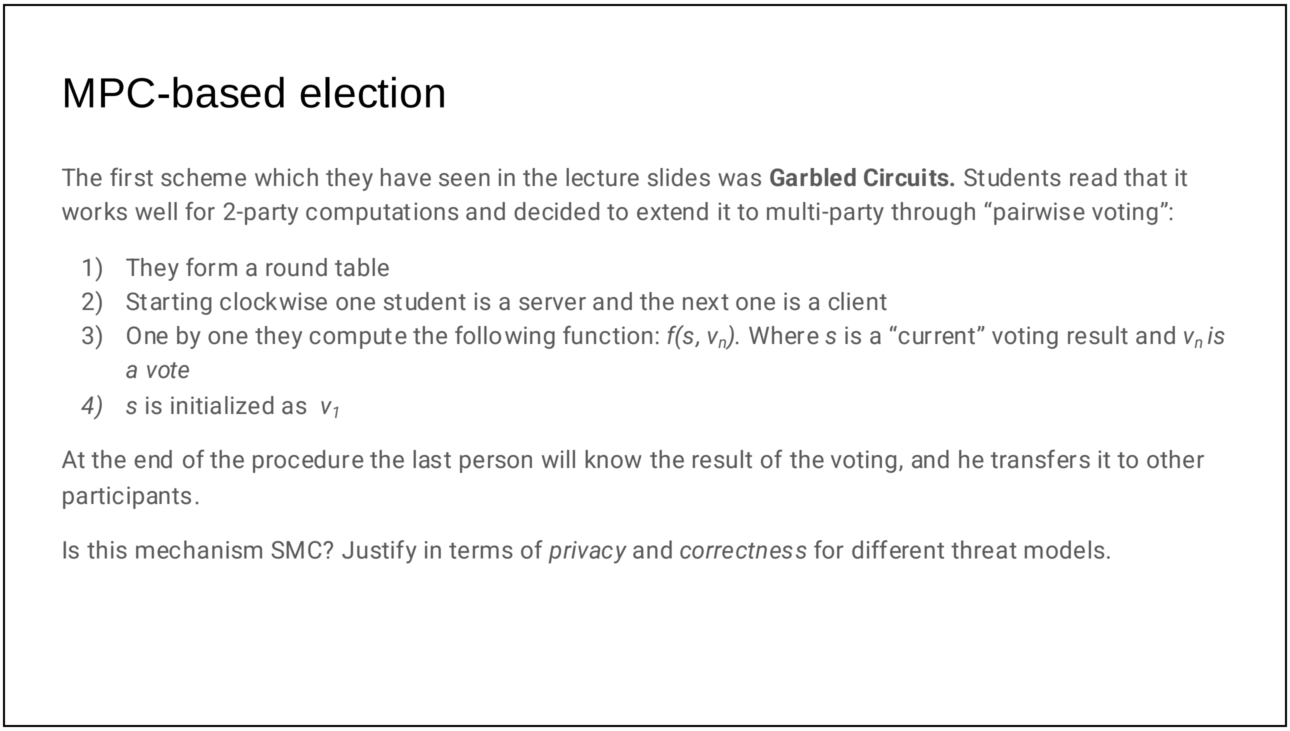


MPC-based election

A small group of CS-523 students decided to elect their student representative. They heard about e-voting and decided to implement e-voting scheme using an SMC protocol.

There are *m* participants, and *n* election candidates. Each vote is an *n*-dimensional one-hot encoded vector with 1 in the position for the selected candidate. The voting outcome is the sum of the vectors.

Example: Suppose there are 4 candidates, so *n = 4*. Then (0, 0, 1, 0) is a vote for the 3rd candidate. The vector (0, 2, 7, 1) could be a result of the election.



MPC-based election

The first scheme which they have seen in the lecture slides was **Garbled Circuits.** Students read that it works well for 2-party computations and decided to extend it to multi-party through “pairwise voting”:

1) They form a round table
2) Starting clockwise one student is a server and the next one is a client
3) One by one they compute the following function: $f(s, v_n)$. Where s is a “current” voting result and v_n *is a vote*
4) s is initialized as v_1

At the end of the procedure the last person will know the result of the voting, and he transfers it to other participants.

Is this mechanism SMC? Justify in terms of *privacy* and *correctness* for different threat models.

It is a warm-up question, just list all the reasons why not.

1) Privacy
• No party should learn anything more than its prescribed output
For example the second one will know the vote of the first one. (True for both honest-but-curious and malicious threat models)
2) Correctness
• Each party is guaranteed that the output that it receives is correct
Not satisfied for malicious threat model.

The followings are not in this year's lecture, but in case students ask for them, we leave the notes here:
3) Independence of Inputs
• Corrupted parties must choose their inputs independently of the honest parties’ inputs
Not satisfied, clients know the voting results of all the participants so far.
4) Guaranteed Output Delivery
• Corrupted parties should not be able to prevent honest parties from receiving their output
5) Fairness
• Corrupted parties should receive their outputs if and only if the honest parties also receive their
outputs

MPC-based election

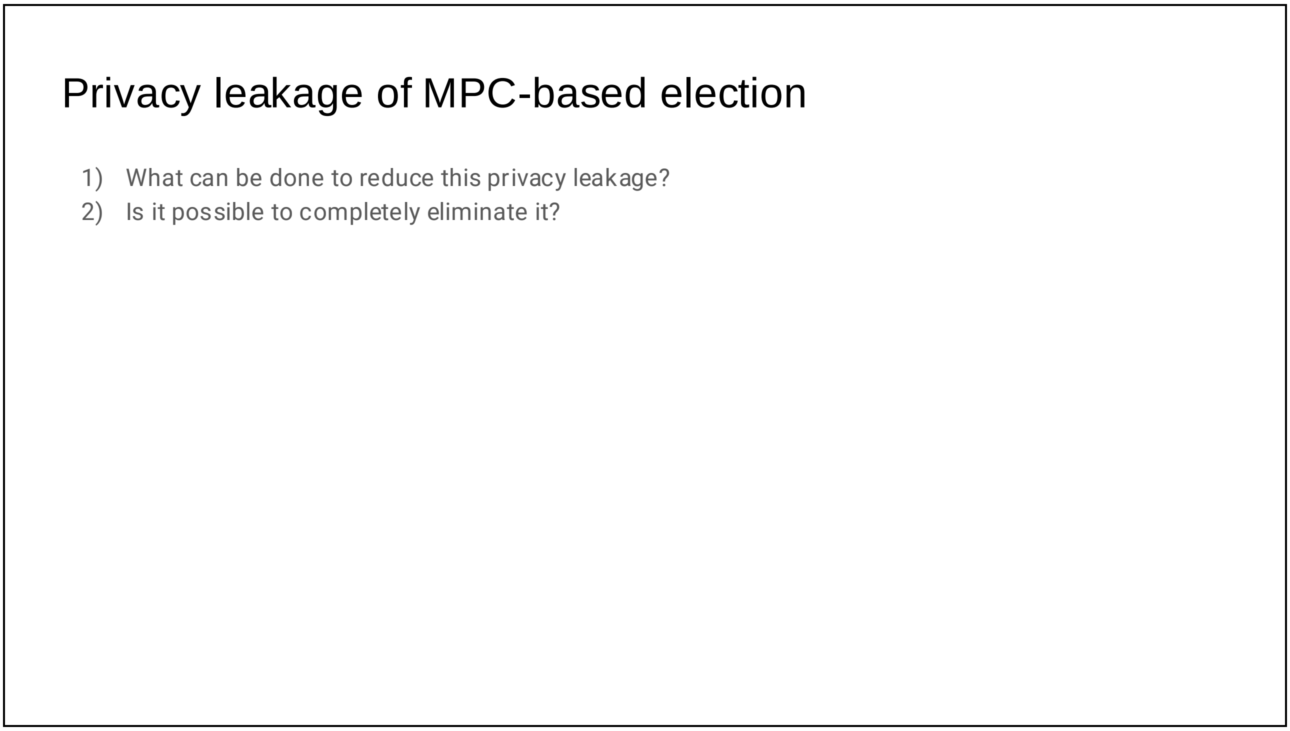
After some discussions they decided to stop inventing a “custom SMC”, and use deployment-ready SMC algorithm in a black-box way (e.g., as SMC based on additive-secret sharing).

- What could be the f(.)?
- Does this scheme guarantee that each participant has zero information about other votes?

Each of them computes $s = f(v_1, ... , v_n) = v1 + ... + vn$

No, two examples:

- A candidate gets zero votes => learn that nobody voted for this candidate
- A candidate that P_i voted for got k votes => learn that k-1 other people voted for this candidate (in the extreme case of two parties, learn other party's vote)



Privacy leakage of MPC-based election

1) What can be done to reduce this privacy leakage?
2) Is it possible to completely eliminate it?

1) Return only number of the candidate who won the election.
2) No, it is not possible, since the functionality (finding out the candidate with the most votes) inherently leaks something about the inputs.