Advanced Tools from Modern Cryptography

Lecture 5

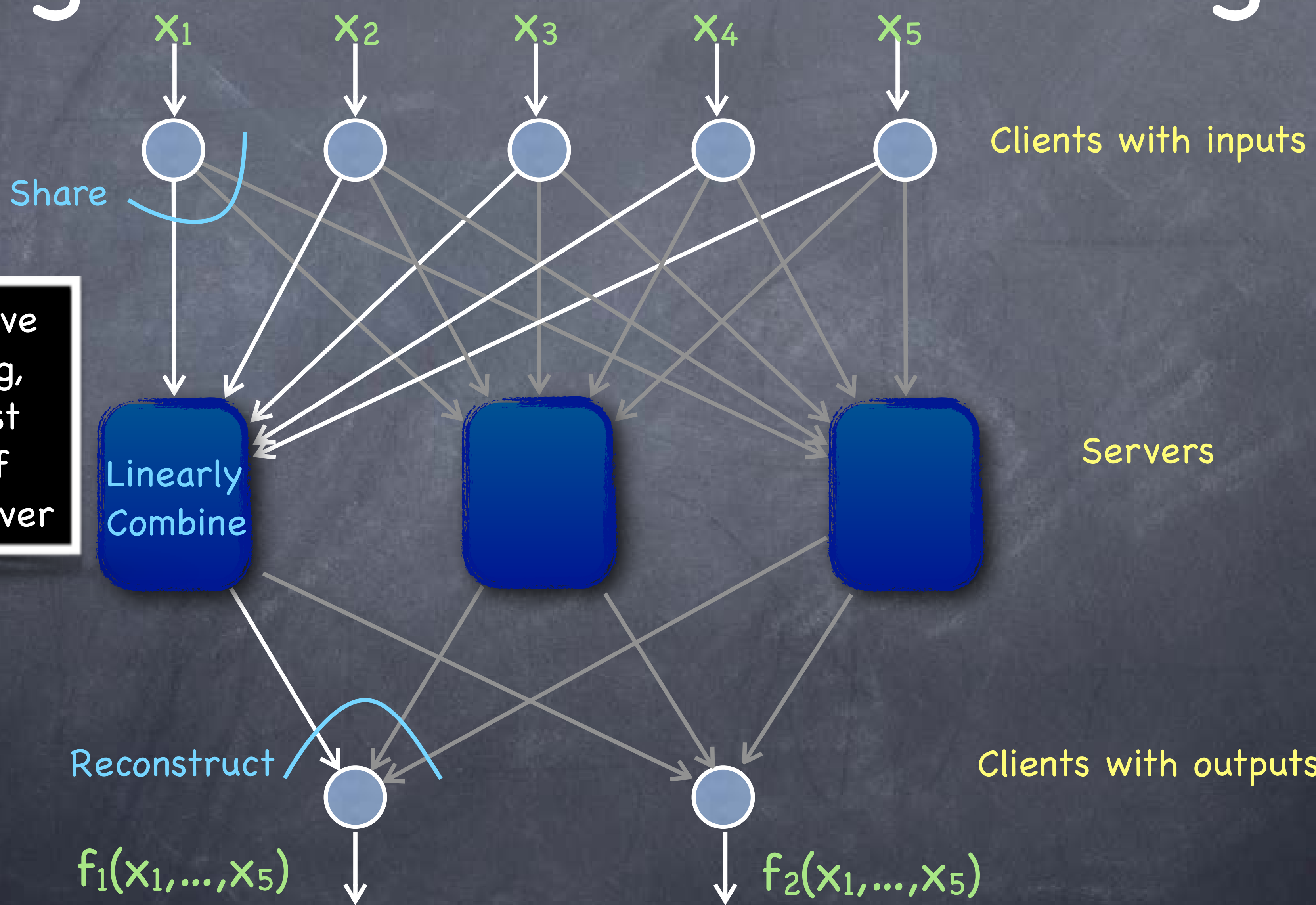
Secure Multi-Party Computation:

Passive Corruption, Honest-Majority, All Functions

Recall

MPC for Linear Functions: Using Linear Secret-Sharing

If using additive secret-sharing, secure against corruption of all-but-one server

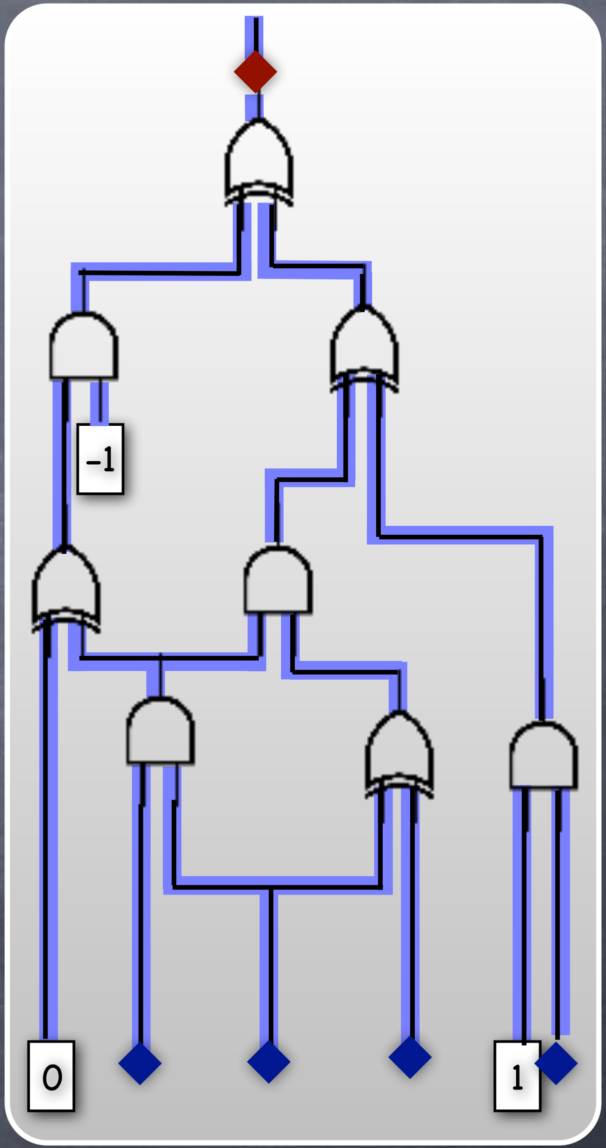


MPC: Honest-Majority + Passive-Corruption

- Today: information-theoretically secure MPC for <u>any</u> function
  - The "BGW protocol" (passive-corruption version)
  - N servers such that adversary can corrupt only < N/2
- Function should be given as an <u>arithmetic circuit</u> over a large enough field (|F| > #parties)
  - Gate-by-gate evaluation, under Shamir secret-sharing of wires

Functions as Circuits

- Directed acyclic graph
  - Nodes: multiplication and addition gates, constant gates, inputs, output(s)
  - Edges: wires carrying values from F
  - Each wire comes out of a unique gate, but a wire might fan-out
  - Can evaluate wires according to a topologically sorted order of gates they come out of



Functions as Circuits

- e.g., Boolean logic as a circuit over GF(2)
  - False = 0, True = 1, $x \wedge y = xy$, $x \oplus y = x + y$
  - $\neg x = 1 + x$, $x \vee y = x + y + xy$
  - e.g.: X > Y for two bit inputs $X = x_1 x_0$, $Y = y_1 y_0$:
    - $(x_1 \wedge \neg y_1) \vee (\neg(x_1 \oplus y_1) \wedge (x_0 \wedge \neg y_0))$

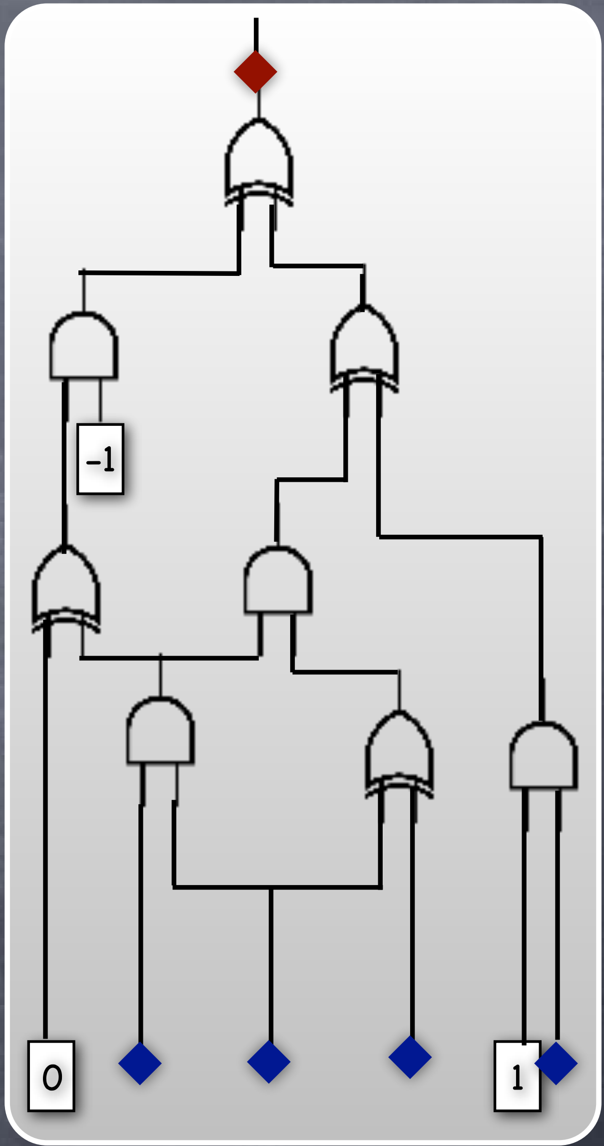
      $= x_1(1+y_1) + (1+x_1+y_1)(1+y_0)x_0$

	00	01	10	11
00	0	0	0	0
01	1	0	0	0
10	1	1	0	0
11	1	1	1	0

- Can directly convert a truth-table into a circuit, but circuit size exponential in input size
- Can convert any ("efficient") program into a ("small") circuit
- Interesting problems already given as succinct programs/circuits

Gate-by-Gate Evaluation

- Wire values will be kept linearly secret-shared among all parties
- Each input value is secret-shared among the servers by the input client "owning" the input gate
- Linear operations computed by each server on its shares, locally (no communication)
  - Shares of x, y → Shares of ax+by
- Multiplication will involve communication
  - Will need appropriate kind of secret-sharing scheme, with threshold < N/2
- Output gate evaluation: servers send their shares to the output client owning the gate



MPC for General Functions: Using Shamir Secret-Sharing

- Question: How to go from shares(x), shares(y) to shares(x·y) securely?
- Idea: Use multiplicative structure of Shamir secret-sharing
  - For polynomials, multiplication commutes with evaluation: $(f \cdot g)(x) = f(x) \cdot g(x)$
  - In particular, to get a polynomial h with $h(0)= f(0) \cdot g(0)$, simply define $h = f \cdot g$. Shares $h(x)$ can be computed as $f(x) \cdot g(x)$
  - But note: h has a higher degree!
    - Problem 1: If original degree $\geq N/2$, can't reconstruct the product even if all parties reveal their new shares
      - Solution: Use degree $d < N/2$ (limits to $d < N/2$ corruption)
    - Problem 2: Can't continue protocol after one multiplication

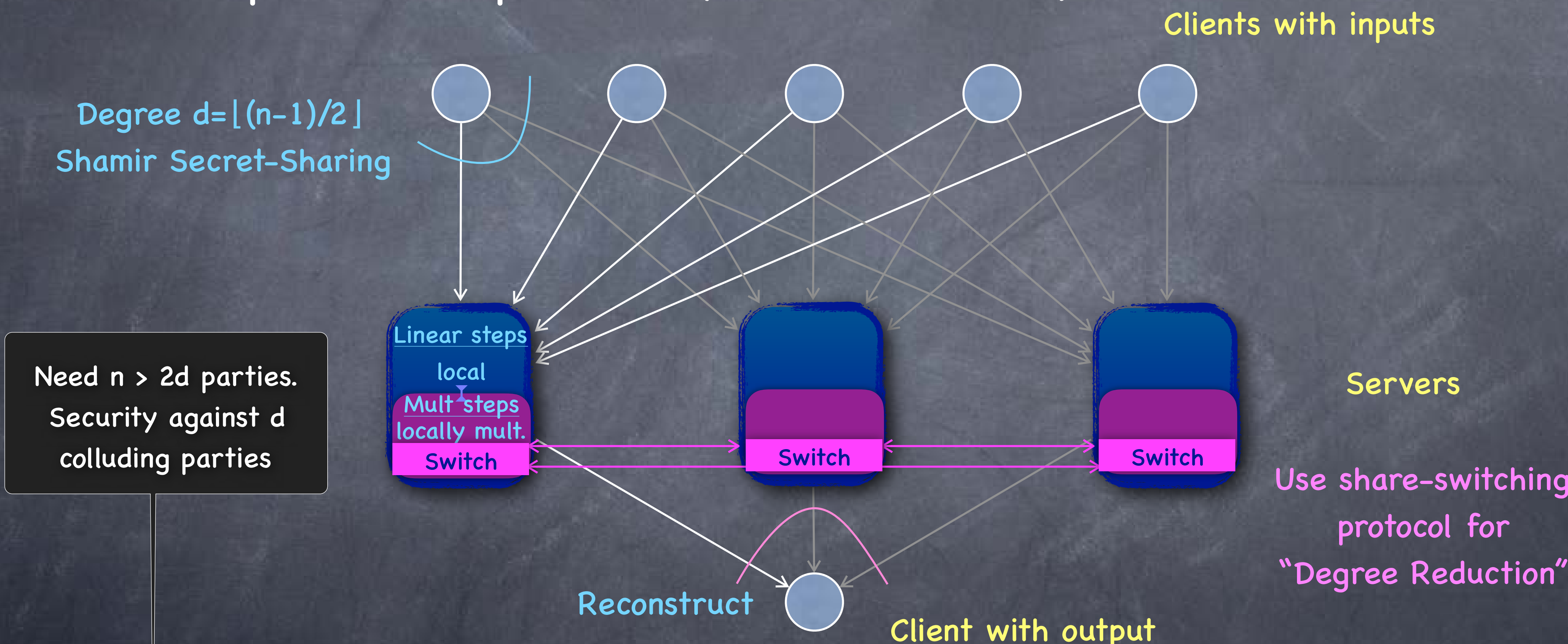
MPC for General Functions: Using Shamir Secret-Sharing

- Problem: If x, y shared using a degree d polynomial, $x \cdot y$ is shared using a degree 2d polynomial
- Solution: Bring it back to the original secret-sharing scheme!
  - Recall share switching: can switch from degree-2d shares to degree-d shares
  - Note: All N servers together should be able to linearly reconstruct the degree-2d sharing
    - Start with $N \geq 2d+1$
    - Can tolerate only up to d ($\leq (N-1)/2$) corrupt servers (and any number of corrupt clients)

      < N/2
  - Security: Exercise (later, using composition)

BGW Protocol for Passive Corruption: Summary

- Function f given as an arithmetic circuit: i.e., a program with linear steps and multiplications (over a finite field)



- Locally multiplying degree d shares gives a degree 2d share
  - Then switch back to a degree d share (involves communicating deg. d shares of deg. 2d shares)

MPC: Honest-Majority + Passive-Corruption

- Typically we consider N parties that can all communicate directly with each other and may have inputs and outputs
  - Each party runs a server (and at most one input and one output client)
- Can compute any N-party function, tolerating corruption of strictly less than N/2 parties
  - e.g., 1 party out of 3, or 2 parties out of 5
  - No security in a 2-party setting!
- Q: For which functions can we obtain information-theoretic security against N/2 (or more) corruption?
  - Not all functions!
  - Exactly known for N=2 (later)
  - General case is still an open problem!

Information-Theoretic MPC Without Honest-Majority?

- Need honest majority for computing AND
- Enough to show that 2 parties cannot compute AND securely
  - Because, if there were an N-party AND protocol tolerating N/2 corrupt parties, we can convert it into a 2-party protocol for AND as follows:
    - Alice runs $P_1,\ldots,P_{N/2}$ "in her head", with her input as P_1's input and 1 as input for the others. Bob runs the remaining parties similarly.
    - View of the parties in Alice's head don't reveal anything about Bob's input, other than what the AND reveals

Information-Theoretic MPC Without Honest-Majority?

- Need honest majority for computing AND
- Enough to show that 2 parties cannot compute AND securely
  - Suppose there is a 2-party protocol for AND. Consider a transcript m such that $Pr[m|x=0,y=0] = p > 0$.
  - By security against Alice, $Pr[m|x=0,y=1] = p$.
    And by security against Bob, $Pr[m|x=1,y=0] = p$.
  - How about $Pr[m|x=1,y=1]$? Should be 0, for correctness
    - Suppose $m=m_1m_2...m_t$, with Alice sending the first message. Alice with x=1 will send m_1 with positive probability because $Pr[m|x=1,y=0] > 0$. Bob with y=1, and given m_1 will send m_2 with positive probability, etc. Hence $Pr[m|x=1,y=1] > 0$!

Today

- Any N-party function can be perfectly securely computed against passive corruption of < N/2 parties
- Linear functions can be perfectly securely computed against the corruption of any number of parties
- There are many functions (e.g., AND) which cannot be information-theoretically securely computed if N/2 parties can be corrupted
- Next: How to go beyond honest-majority (hint: not information-theoretically)