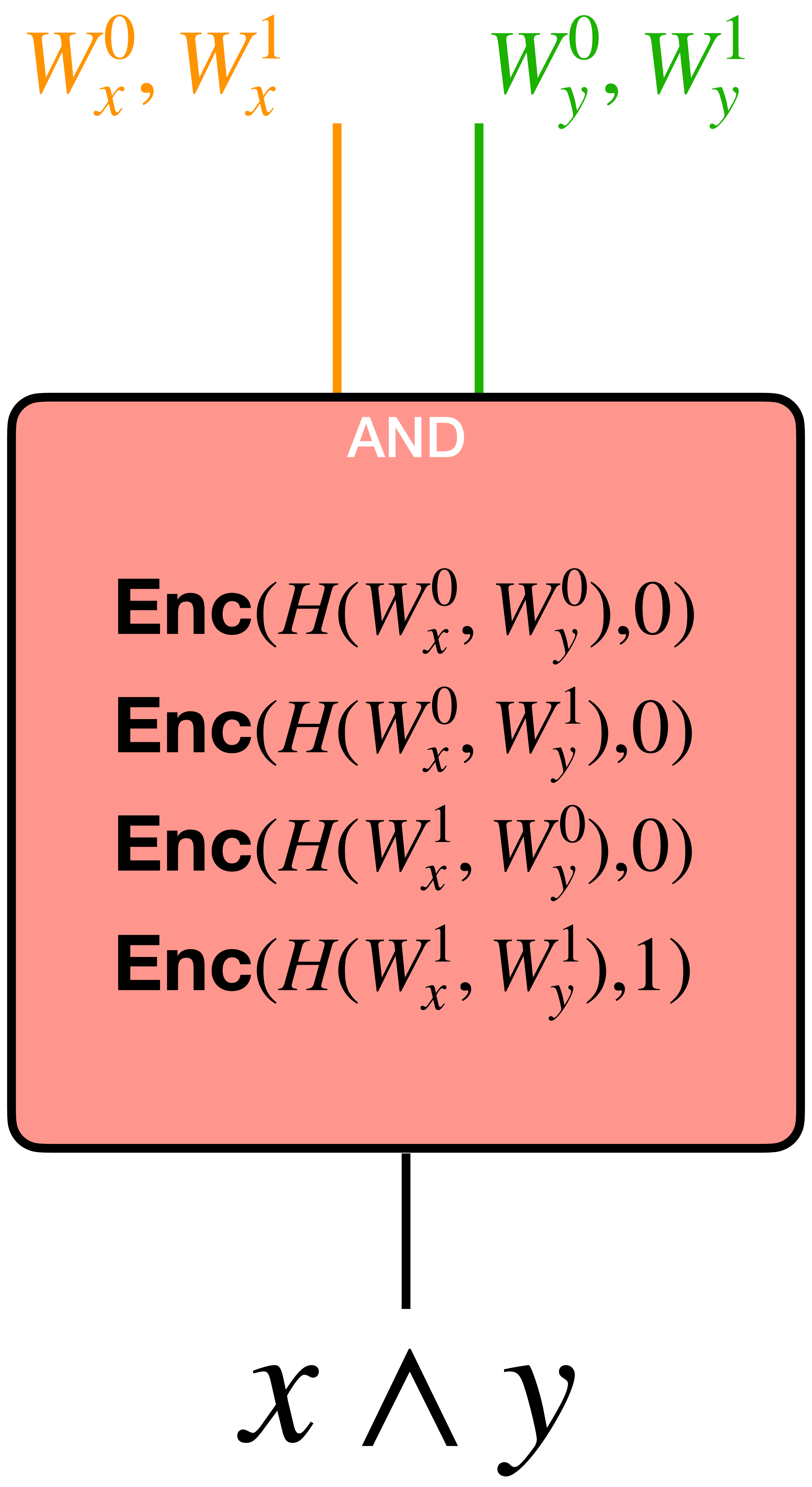
Maliciously secure MPC

Garbled circuits

- Let H be a key derivation function
- Pick four random labels: $W_x^0, W_x^1, W_y^0, W_y^1$, which correspond to the four possible values for x and y
- For each row
  - Use H to derive a key using the corresponding labels
  - Encrypt the result
- Randomly permute the rows


W_x^0, W_x^1
W_y^0, W_y^1
AND
Enc$(H(W_x^0, W_y^0),0)$
Enc$(H(W_x^0, W_y^1),0)$
Enc$(H(W_x^1, W_y^0),0)$
Enc$(H(W_x^1, W_y^1),1)$
$x \wedge y$

Malicious adversary

- What can a malicious adversary do?
  - Give fake inputs (not something MPC can handle)
  - Give inconsistent inputs
  - Execute incorrect computation
  - Refuse to release the final result (cannot always be handled)

Maliciously secure GC via cut-and-choose

- P_0 is the garbler, P_1 is the evaluator
- Problem: P_0 can generate an incorrect garbled circuit (perhaps an identity circuit that embeds P_1's input!)
- Idea:
  - P_0 generates many circuits
  - P_1 asks P_0 to open some of them (cut), then evaluates the rest (choose)
  - P_1 takes the majority output
- Why not abort if the outputs are inconsistent?
  - Insecure! The incorrect circuits can be *selectively incorrect* based on P_1's input, e.g., gives wrong answer if the first bit is 1

Maliciously secure GC via cut-and-choose

- Optimal choice in cut-and-choose
  - P_1 checks e circuits and opens $(s-e)$ circuits
  - Probability that P_0 cheats without getting caught: $\binom{s-b}{s-e} / \binom{s}{s-e}$
  - P_0's best strategy is to construct $b = \lfloor e/2 \rfloor + 1$ bad circuits
- Some concrete numbers
  - LP08: if open $s/2$ circuits, then $2^{-0.311s}$ probability of failure, need 128 to achieve security 2^{-40}
  - SS11: open around 60% circuits, need 125 circuits for 2^{-40} security

Problem: input consistency

- Ensuring P_1's input is easy: P_0 batches the corresponding labels across circuits, run OT once per P_1's input bit, for the entire batch
- But P_0 could give inconsistent input labels across different circuits
- Primitives:
  - Commitment Com(x): commit to a chosen value; hiding & binding
  - Universal hash function: A collection of has functions $\mathscr{H} = \{h \mid h : A \to B\}$ is universal if for any distinct x, y, the probability that a uniformly chosen h satisfies $h(x) = h(y)$ is at most $1/|B|$

Input consistency check

- Input consistency check protocol:
    - P_0 commits to input labels $x^1, x^2, \cdots, x^s$ where $x^j = \text{Encode}(x \| r)$ are the inputs to the j-th circuit and sends commitments to P_1
    - P_0 and P_1 jointly and uniformly pick $h \in \mathscr{H}$
    - P_0 constructs s copies of the circuit, encoding both the function and an auxiliary circuit that computes $h(x^j)$
    - P_1 checks a random subset of the circuits; if check passes, P_0 decommits input values for the rest of the circuit
    - P_1 first evaluates the auxiliary circuits. If the hashes are consistent, then evaluate the remaining objective circuits

Input consistency check

- Security: how can P_0's inputs be inconsistent?
  - Auxiliary circuits are faulty -> cut-and-choose ensures non-faulty circuits
  - P_0 has found a collision to the hash function -> security by definition of hash function
  - P_0 can break the commitment scheme -> Com(x) assumed to be binding
- The randomness r input by P_0 ensures hiding (via left-over-hash lemma)

Problem: selective abort

- P_0 cheats by providing *incorrect* labels in OT
    - Select inputs to OT such that when P_1's first bit is 1, then it will get garbage label, causing an abort since the circuit cannot be evaluated
- Idea: encode P_1's true input into an alternative input such that the joint distribution of a subset of the input is uniform (independent of P_1's true input)
    - <u>Simple solution</u>: secret share each input bit into s random bits (total ns bits)
    - <u>Fancier solution (fewer generated bits)</u>:
        - Given y, compute new input $\bar{y}$ such that $y = M \cdot \bar{y}$ where M is k-probe-resistant
        - Can instantiate with Reed-Solomon code

Today’s reading: maliciously secure collaborative analytics

Next class: guest lecture

Title: Law and Policy for the Quantum Age

Abstract: Quantum technologies are so different from our conventional intuition that they seem like science fiction—yet some quantum technologies can be commercially purchased today, and more are just around the corner, posing profound policy issues. For example, quantum sensing arrays might someday be trained to recognize weapons or sniff the molecules of contraband, even if concealed, or detect guns in private homes by measuring electromagnetic or gravitational fields through roofs. Quantum algorithms will speed up the process of cracking the encryption that protects our communications. Quantum simulation will have tremendous benefits for the environment, but these same techniques could be used to engineer more powerful biological, chemical, synthetic, conventional, and even genetic weapons. Berkeley Professor Chris Hoofnagle, who has a forthcoming book on The Quantum Age with Simson Garfinkel (Cambridge Univ. Press 2021), will describe the state of the science in quantum technologies and consider the policy consequences of the quantum age.