Why, in the semi-honest model, is it required to define security for probabilistic functionalities by comparing the *joint* distribution (that includes the honest party's input, to the distribution in the simulated execution)?

We considered in class the functionality in which **A** outputs a random bit and **B** outputs nothing. **B** should clearly not learn **A**'s output bit.
We showed a protocol that is clearly insecure: **A** chooses a random bit, outputs it, and sends the bit to **B** (who ignores it). This protocol is insecure since B learns A's output.

We could have proved this protocol to be secure if we used a security definition that compares the view of a corrupt B in the real execution to the output of a simulator that only gets B's input and output: The simulator should generate a transcript that contains a single random bit sent from A to B.
This demonstrates that a security definition that does not take into account the joint distribution of *both* parties is insufficient. Note that the protocol cannot be proved to be secure according to the security definition that looks compares the joint definition: In the real execution the bit sent to B is identical to the bit output by A. In the simulation the simulator does not have access to A's output and therefore the value that it generates for B's transcript would be independent of A's output.