Commitments

Jackson Mowry

Thu Oct 2 08:25:58 2025

1. What is computational binding and hiding?

   - Binding describes the phenomenon that only 1 chosen value can produce the associated commitment. If there are a range of values that can be chosen to produce the same commitment then the user providing the commitment is not actually **bound** to the value they originally chose.
   - Hiding is describing the fact that the actual chosen value (the committed value) cannot be known simply by inspecting the commitment as a whole.

2. What is perfect binding and hiding?

   - The perfect binding property would mean that only a single value can produce the associated commitment, with 0 other possible values producing an equal commitment.
   - The perfect hiding property would mean that there are infinitely many possible inputs that could product the same output, meaning there is no way to know what the original committed value was, even if enumerating all possible inputs was achievable.

3. Is it possible to have a scheme that has both perfect binding and hiding? Why or why not?

   - No this is impossible. If we look at the properties described above they are at odds with each other. Improving hiding necessarily means weakening binding, and vis-versa. If the committing party has infinitely many options to choose from that produce the same output commitment they can change their committed value after sending the original commitment. In the same way if there is only one (or very few) inputs that can product the same output commitment the receiving party can simply enumerate all possible inputs to know what values were committed to.

4. For the following protocols from the class slides, identify whether they have computational or perfect binding and hiding
   (a) Hash-based commitment
      - Perfect hiding
         - We're generating a random number alongside our committed value, meaning there are multiple x/r values that give the same output.
         - Additionally both x and r values are equally likely to take any value
         - This is all dependent on us having a perfect hash function that does not reveal anything about its inputs
      - Computationally binding given that the output of a hash function is large enough that finding a collision in hashes in infeasible, i.e. it is resistant to a pre-image attack

   (b) Pedersen commitment scheme
      - Perfect hiding
      - Computationally binding because the discrete log problem is believed to be difficult to solve