

CS 395T - Theory and Practice of Secure Systems
Fall 2006

Homework #3

Due: 3:30pm CST (in class), November 21, 2006

NO LATE SUBMISSIONS WILL BE ACCEPTED

YOUR NAME: _____

Collaboration policy

No collaboration is permitted on this assignment. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at <http://www.cs.utexas.edu/users/ear/CodeOfConduct.html>

Homework #3 (30 points)

Problem 1 (5 points)

Given any 1-out-of-2 oblivious transfer protocol OT_1^2 , use it to construct a 1-out-of-4 oblivious transfer protocol OT_1^4 .

The OT_1^2 protocol must be used as a “black box,” *i.e.*, participants in OT_1^4 are not allowed to modify OT_1^2 , they can only call it on some inputs. (Imagine that each participant in OT_1^4 is given two black boxes, one enabling him to play the role of the chooser in an instance of OT_1^2 , the other enabling him to play the role of the sender.)

Problem 2

Recall Schnorr's ID protocol, which is an honest-verifier zero-knowledge proof of knowledge of the discrete logarithm s of $t \pmod p$, where p is a large prime, and t is a generator of an order- q subgroup. Let P be the prover, V the verifier.

$$\begin{array}{ll} P \rightarrow V & x = g^r \pmod p \quad \text{where } r \text{ is a random value between } 1 \text{ and } q - 1 \\ P \leftarrow V & c \quad \text{where } c \text{ is a random } k\text{-bit value} \\ P \rightarrow V & y = sc + r \pmod q \end{array}$$

Verifier accepts the proof if $x \cdot t^c = g^y$.

Problem 2a (2 points)

Suppose Larry and Moe execute the above protocol with Larry acting as the prover and Moe acting as the verifier. Now Moe wants to convince Curly that he knows the discrete logarithm of t . He records the transcript of his conversation with Larry and gives it to Curly.

Should Curly be convinced that Moe knows the discrete logarithm of t ? Explain.

Problem 2b (6 points)

Suppose that instead of sending his messages directly to Moe (the verifier), Larry (the prover) signs his messages and gives them to Curly, who forwards them to Moe. Similarly, Moe signs his messages and gives them to Curly, who forwards them to Larry.

Assume that they are using an unforgeable digital signature scheme, *i.e.*, it is not feasible for Curly to forge Larry's (respectively, Moe's) signature on a message that Larry (respectively, Moe) did not sign. Also assume that Larry knows Moe's public signature verification key, and vice versa.

Should Moe be convinced that Larry knows the discrete logarithm of t ? Explain.

Should Curly be convinced that Larry knows the discrete logarithm of t ? Explain.

Problem 3 (6 points)

Let (N, e) be Larry's RSA public key, and d the corresponding private key. Both Moe and Curly know a ciphertext c encrypted under Larry's public key. Moe claims that he knows the corresponding plaintext m (recall that with RSA, $m = c^d \pmod N$), and that he can prove this to Curly without revealing m .

He does it via the following protocol:

- Moe generates a random number $r \pmod N$, encrypts it under Larry's public key to obtain $x = r^e \pmod N$, and sends x to Curly.
- Curly flips a fair coin.

Coin comes up heads: Curly asks Moe to send him the plaintext of x , *i.e.*, $x^d \pmod N$. Moe sends him $y = r$. Curly verifies the answer by checking $x = y^e \pmod N$.

Coin comes up tails: Curly asks Moe to send him the plaintext of $c \cdot x$, *i.e.*, $(c \cdot x)^d \pmod N$. Moe sends Curly $y = m \cdot r$. Curly verifies the answer by checking $x \cdot c = y^e \pmod N$.

Prove that this protocol is zero-knowledge even if Curly is malicious. Hint: it is sufficient to write a simulator that with probability $\frac{1}{2}$ creates a transcript which is indistinguishable from the transcript of Moe and Curly's conversation.

Problem 4 (6 points)

Sergey Brin and Michael Dell want to find out who is richer. For simplicity, assume that each has X billion of dollars, where X is some integer between 1 and 8. Can they use Yao's "garbled circuits" protocol, and, if so, how?

For full credit on this problem, write down the entire protocol, including all garbled truth tables.

Problem 5 (5 points)

Imagine a flawed key establishment protocol, in which the adversary learns the parity bit of the key (*i.e.*, sum of all bits modulo 2) by observing protocol messages. Demonstrate that this protocol is unsimulatable in the ideal key exchange functionality by explaining what action can be performed by the real-world adversary so that the resulting transcript cannot be simulated in the ideal world.