CS355: Topics in cryptography **Spring 2000**

Assignment #2

Due: Wednesday, May 31st, 2000.

Problem 1: Let p be a prime and $g \in \mathbb{Z}_p^*$ be an element of prime order q. A Diffie-Hellman tuple is a 4-tuple of the form $\langle g, g^a, h, h^a \rangle$ in $\mathbb{Z}_p$. Consider the following proof system for proving that a tuple $\langle g, u, h, v \rangle$ is a DH tuple: (we let $u = g^a$ and $v = h^a$).

Step 1: The verifier commits to a challenge $c \in \mathbb{Z}_q$.

Step 2: The prover picks a random $r \in \mathbb{Z}_q$ and sends $y_1 = g^r$ and $y_2 = h^r$ to the verifier.

Step 3: The verifier opens the commitment and reveals c to the prover.

Step 4: The prover responds with $z = ac + r \bmod q$.

Step 5: The verifier accepts if and only if $g^z = u^c \cdot y_1$ and $h^z = v^c \cdot y_2$.

Answer the following questions:

a. Prove that this protocol is zero knowledge. Prove completeness, soundness, and zero knowledge. You may assume that the commitment in Step 1 is secure against an infinitely powerful receiver (in this case, the prover). In your proof of soundness state what is the maximum probability that a malicious prover fools the verifier into thinking that a non-DH tuple is a DH-tuple.

b. Prove that repeating this protocol n times in *parallel* to prove that n tuples $\langle g_i, u_i, h_i, v_i \rangle$ are DH tuples is still a zero-knowledge protocol. As we said in class, parallel repetition generally does not preserve zero-knowledge. However, in this case it does.

c. Show that if the order of g in $\mathbb{Z}_p^*$ has small prime divisors then the protocol is no longer sound. For instance, if 3 divides the order of g then a malicious prover has a good chance of fooling the verifier.

Problem 2: Let $N = pq$. Consider the following protocol for proving that x is a quadratic residue in $\mathbb{Z}_N$: (let $x = \alpha^2 \bmod N$).

Step 1: The prover picks a random $r \in \mathbb{Z}_N$ and sends $y = r^2 \bmod N$ to the verifier.

Step 2: The verifier picks a random $b \in \{0, 1\}$ and sends b to the prover.

Step 3: The prover responds with $z = b \,?\, \alpha \cdot r \,:\, r$.

Step 4: The verifier accepts if and only if $z^2 = y \cdot x^b \bmod N$.

In class we showed that this protocol is zero-knowledge. Show that the protocol is a zero-knowledge proof of knowledge. In other words, construct an extractor for α from a prover that always succeeds in convincing the verifier.

Problem 3: We consider the problem of composition of protocols.

a. Let P_1, P_2 be zero-knowledge protocols for NP-statements T_1, T_2 respectively. The sequential composition of P_1 and P_2 is the protocol P_1 followed by the protocol P_2. Is the sequential composition of P_1, P_2 a zero-knowledge proof for the statement T_1 and T_2? This is not known to be true. Explain why.

b. Show that zero-knowledge is maintained under a sequential composition of *black-box* zero-knowledge protocols. Recall that *black-box* zero-knowledge is weaker than general zero-knowledge in that a single simulator S must simulate the interaction with an arbitrary verifier V^*. The simulator is only allowed to use the verifier V^* as a black-box. In other words, the simulator can only run V^*, or rewind V^*. It cannot look at how V^* is implemented. All the examples of zero-knowledge protocols we have so far are black-box zero-knowledge protocols.

c. Let P_1, P_2 be witness indistinguishable proofs for T_1, T_2 respectively. We assume P_1, P_2 use the same number of rounds m. A parallel composition of P_1 and P_2 is the protocol P where the i'th round of P is the concatenation of the i'th round of P_1 and P_2. The protocol P uses m rounds. Show that P is a witness indistinguishable proof for the statement $T_1 \text{ and } T_2$. We note that witness indistinguishability is maintained under any interleaving of P_1 and P_2.

Problem 4: (extra credit) Recall that in class we showed how to build a bit-commitment scheme out of any one-way permutation (simply use a hard-core predicate of the function). The scheme is secure against an infinitely powerful committer and a bounded verifier. We also showed a scheme based on discrete-log with the opposite property: the scheme is secure against an infinitely powerful verifier and a bounded committer (recall: $\mathsf{commit}(x) = g^x \cdot h^r \bmod p$ for a random r). Show that one can build a commitment scheme with the second property out of any one-way permutation (rather than specifically using the discrete log one way permutation which is relatively slow).