CSE 208: Advanced Cryptography Fall 2002

Problem Set 1

Instructor: Daniele Micciancio Oct. 17, 2002

Due: Thu. Oct. 31, 2002

1 Zero-Knolwedge and Simulators

In class we specified that a proof system (P, V) for a language L is Zero Knowledge if for every (possibly cheating) probabilistic polynomial time verifier V^*, there exists a probabilistic polynomial time simulator S such that for every input in the language $x \in L$, and every (polynomial size) auxiliary input a, the view of the cheating verifier $View_{V^*}[P(x), V^*(x, a)]$ is computationally indistinguishable from the output of the simulator $S(x, a)$.

A similar simulator based technique was used to define secure computation of general functions, e.g., the security of the sender in the oblivious transfer protocol. The basic ideas of this problem applies to general secure computation protocols, but for simplicity we will concentrate on zero knowledge proofs, as a representative example.

A question that arose in class on several occasions is: how does the simulator depend on the verifier? We answered this question saying that "typically" there is a polynomial time computable transformation that on input a description (the "code") of the verifier V^*, outputs a description of the corresponding simulator S. So, consider this alternative (seemingly stronger) definition of zero knowledge: a prook system (P, V) is "transformation" zero knowledge (XZK) if there is a polynomial time computable function T such that for any probabilistic polynomial time interactive program V^*, outputs another probabilistic polynomial time interactive program $S = T(V^*)$, such that for every input in the language $x \in L$, and every (polynomial size) auxiliary input a, the view of the cheating verifier $View_{V^*}[P(x), V^*(x, a)]$ is computationally indistinguishable from the output of the simulator $S(x, a)$.

As a special case, we noted that the simulator can use the verifier in a black-box manner, and this will be the case for most proof systems used in this course. Specifically we gave the following definition: a proof system (P, V) is black-box zero knowledge (BZK) if there is a probabilistic polynomial time oracle machine $S^{(\cdot)}$ such that for any probabilistic polynomial time cheating verifier V^*, and for every input in the language $x \in L$, $View_{V^*}[P(x), V^*(x)]$ is computationally indistinguishable from the output of the simulator $S^{(V^*)}(x)$, when given black-box access to V^*. (As a reminder, black-box access means that S can interact with V^*, possibly fix the value of the random tape of V^*, restart the program V^* multiple times, but it does not have access to the actual code of V^*.)

Notice that in the definition of BZK we omitted the auxiliary input, and said several times in class that for BZK the auxiliary input is not necessary.

In this problem you are asked to compare these notions of zero knowledge.

Part (a) [5 points] Show that ZK is equivalent to XZK. The implixation $XZK \Rightarrow ZK$ is trivial, the interesting direction is the opposite one: show that if for every cheating verifier there exists a simulator, then there is also an efficient transformation that on input the code of the cheating verifier, outputs a corresponding simulator. [Hint: this implication relies in a substantial way on the use of the auxiliary input, i.e., if no auxiliary input in included the implication is not necessarily valid.]

Part (b) [5 points] Show that BZK implies ZK. Here the main difficulty to be overcome is that the definition of BZK does not include an auxiliary input.

Part (c) [1000 points!] Show that ZK does not implies BZK, i.e., there are proof systems that satisfies definition ZK, but not BZK. I do not really expect anybody to answer this, I just wanted to let you know that BZK is a seemingly stronger notion than ZK and XZK.

2 String Oblivious Transfer

Say we want to build an $\binom{2}{1} - OT^l$ protocol where the messages of the sender are l-bit strings. In class we gave a solution for the case $\binom{2}{1} - OT^1$ of single bit messages. A possible solution (for semihonest receiver) for $l > 1$ is to repeat the bit OT protocol l times, and then enforce honest behaviour adding a ZK proof that the verifier always requested the same message in all transfers. (This should be done in a careful way, as if the proof is performed only at the end, the sender might find out that the verifier was cheating when it is too late.)

This solution has the undesirable effect that the communication overhead grows lienarly with the bit length l. In this problem we consider a more efficient solution directly inspired to the protocol presented in class. For simplicity, we consider the case of strings of length 2.

This is the protocol: (The input of the sender is $v_{11}v_{12}, v_{21}v_{22}$, and the input of the receiver c, where all variables are single bits. The receiver should ger $v_{c1}v_{c2}$, but nothing more, and the sender should not learn c. The formal definition is essentially the same as the one given in class.)

1. The sender chooses a pair (f, t) from a family of trapdoor permutations with hard code predicate B, and send f to the receiver.
2. The receiver chooses x and $y_{\bar{c}}$ at random (from the domain of f, and computes $y_c = f(f(x))$. The values y_0, y_1 are sent to the sender.
3. The sender computes $u_{ij} = v_{ij} \oplus B(f^{-j}(y_i)$ for $i, j = 1, 2$, and send them all to the receiver.
4. The receiver outputs $u_{c1} \oplus B(f(x))$ and $u_{c2} \oplus B(x)$.

Part (a) [1 point] Show that the protocol is correct, i.e., if all parties follow the protocol, the the receiver outputs the right messages.

Part (b) [2 points] Show that the protocol is secure for the receiver, i.e., for every possibly cheating sender, the views of S' in $[S', R(0)]$ and $[S', R(1)]$ are computationally indistinguishable. (In fact, they should be identical.)

Part (c) [5 points] Show that the protocol is secure for the sender, if the receiver is semihonest.

Part (d) [2 point] Show how to transform the above protocol to one that is secure with respect to malicious attacks. The transformation and proof of security, using ZKPK, is essentially the same as the one described in class. You are not required to repeat the proof of security. Just give a description of the protocol, including a clear definition of the NP language and underlying NP relation for which the ZKPK is given.

3 1 out of many OT

In this problem we consider another possible extension of OT protocols. Instead of selecting 1 value out of 2, the receiver chooses 1 value out of k possible ones.

Consider the following protocol for $\binom{3}{1} - OT^l$. Assume a $\binom{2}{1} - OT^l$ protocol is given, (S_2, R_2). We use it as a subprotocol to define a protocol (S_3, R_3). The input to S_3 is a triplet of values v_0, v_1, v_2, where each v_i is a string of length l. The input to R_3 is an index $i \in \{0, 1, 2\}$.

1. The sender choose a string w of length l uniformly at random, and then invokes $S_2(v_0, w)$ and $S_2(v_1 \oplus w, v_2 \oplus w)$.
2. If $i = 0$, the receiver invokes $R_2(0) = z_1, R_2(0) = z_2$ and outputs z_1.
3. If $i = 1$, the receiver invokes $R_2(1) = z_1, R_2(0) = z_2$ and outputs $z_1 \oplus z_2$.
4. If $i = 2$, the receiver invokes $R_2(1) = z_1, R_2(1) = z_2$ and outputs $z_1 \oplus z_2$.

Part (a) [1 points] Show that the protocol (S_3, R_3) is correct.

Part (b) [3 points] Show that is (S_2, R_2) is secure for the receiver, then also (S_3, R_3) is secure for the receiver, i.e., for any two indices i, j, and any possibly cheating sender S, distributions $View_S[S, R_3(i)]$ and $View_S[S, R(j)]$ are indistinguishable.

Part (c) [5 points] Show that if (S_2, R_2) is secure for the sender, then also (S_3, R_3) is secure for the sender, i.e., for any possibly cheating receiver R, there is simulator (Q, Z) such that for any v_0, v_1, v_2 and auxiliary information a, $Z(a, v_{Q(a)})$ is computationally indistinguishable from $View_R[S_3(v_0, v_1, v_2), R(a)]$.

Part (d) [1 point] Show how the previous protocol can be generalized to $\binom{k}{1} - OT$. (Both the protocol and proof of security generalize the one for (S_3, R_3). You should prove the correctness and security of the protocol on your own to make sure your protocol is good, but no formal proof is required for the homework.)

4 OT and secure computation

In class we gave a ad-hoc definition of security for $\binom{2}{1} - OT$ protocol (S, R) as follows:

- (Security for the receiver, $R - sec - OT$) For every possibly cheating sender S', distributions $View_{S'}[S', R(0)]$ and $View_{S'}[S', R(1)]$ are computationally indistinguishable (with respect to non uniform polynomial circuits).
- (Security for the sender, $S - sec - OT$) For every possibly cheating receiver R', there is a simulator (Q, Z) such that for any inputs v_0, v_1, c, $View_{R'}[S(v_0, v_1), R'(a)]$ is computationally indistinguishable from the output of $S(a, v_{Z(a)})$.

An alternative way to define OT is to consider it as a special case of secure two party computation protocol, where the function to be computed is $f((v_0, v_1), c) = (\perp, v_c)$. Here, we define an ideal protocol where the idealized sender IS sends (v_0, v_1) to a trusted party T, and the idealized receiver IR sends c to T, and in turns T sends v_c to IR and $\perp$ to IS. Following the general definition of security, we say that (S, R) is a secure OT protocol if

- (Security for the receiver, $R - sec - MPC$) For any adversarial sender A, there is an ideal adversary IA such that for any input c and auxiliary input a, the joint output of $Exec[A(a), R(c)]$ is indistinguishable from the output of $Ideal[IA(a), IR(c)]$.
- (Security for the sender, $S - sec - MPC$) For any adversarial receiver A, there is an ideal adversary IA such that for any input v_0, v_1 and auxiliary input a, the joint output of $Exec[S(v_0, v_1), A(a)]$ is indistinguishable from the output of $Ideal[IS(v_0, v_1), IA(a)]$.

In this problem we study the relation between these two definitions.

Part (a) [1 point] Prove that if (S, R) is $S-sec-MPC$ secure, then it is also $S-sec-OT$ secure.

Part (b) [2 point] Prove that if (S, R) is $S-sec-OT$ secure, then it is also $S-sec-MPC$ secure.

Part (c) [2 point] Prove that if (S, R) is $R-sec-MPC$ secure, then it is also $R-sec-OT$ secure.

Part (d) [5 point] Prove that if (S, R) is $R-sec-OT$ secure, then it is also $R-sec-MPC$ secure.