CSCI 2590: Advanced Topics in Cryptography	Feb 7, 2019
Problem Set 1	
Instructor: Anna Lysyanskaya	*Due: Feb 21, 2019*

Problem 1: Zero-Knowledge in Parallel, Fixed

Let (GEN, COM, VER) be a perfectly hiding commitment scheme with the following properties:

Correctness: For all security parameters k and inputs α,

$$\Pr[\mathsf{pp} \leftarrow \mathrm{GEN}(1^k); (c,d) \leftarrow \mathrm{COM}(\mathsf{pp},\alpha) : \mathrm{VER}(\mathsf{pp},c,d,\alpha) = \mathsf{True}] = 1$$

Binding: For all k and for any *probabilistic polynomial-time* cheating commiter C^*:

$$\Pr\left[\begin{array}{l}\mathsf{pp} \leftarrow \mathrm{GEN}(1^k); (c,d_1,d_2,\alpha_1,\alpha_2) \leftarrow C^*(\mathsf{pp},\alpha) : \\ \mathrm{VER}(\mathsf{pp},c,d_1,\alpha_1) = \mathrm{VER}(\mathsf{pp},c,d_2,\alpha_2) = \mathsf{True} \text{ and } \alpha_1 \neq \alpha_2\end{array}\right] < \text{negligible}(k)$$

Perfect Hiding: For all k and and all inputs α and β, the following distributions are identical:

$$\left\{\mathsf{pp} \leftarrow \mathrm{GEN}(1^k); (c,d) \leftarrow \mathrm{COM}(\mathsf{pp},\alpha) : (\mathsf{pp},c)\right\} = \left\{\mathsf{pp} \leftarrow \mathrm{GEN}(1^k); (c,d) \leftarrow \mathrm{COM}(\mathsf{pp},\beta) : (\mathsf{pp},c)\right\}$$

Here we provide a five round proof system for the following language on pairs of graphs:

$$L = \{(G,H) \mid G \text{ is isomorphic to } H\}$$

1. The prover selects $\mathsf{pp} \leftarrow \mathrm{GEN}(1^k)$ and sends pp to the verifier.
2. The verifier chooses a k-bit random string r, selects $(c,d) \leftarrow \mathrm{COM}(\mathsf{pp},r)$ and sends c to the prover.
3. The prover randomly selects k graphs $C_1,\ldots,C_k$ such that each C_i is isomorphic to G and sends $C_1,\ldots,C_k$ to the verifier.
4. The verifier sends d and r to the prover.
5. If $r = \mathrm{VER}(\mathsf{pp},c,d)$ then for each graph C_i the prover sends the verifier a random isomorphism mapping G to C_i if the i th bit of r is 0 and a random isomorphism mapping H to C_i if the i th bit of r is 1.

Prove that the above protocol has negligible soundness and is zero-knowledge.

Problem 2: Tying loose ends from Feige-Shamir

We have two loose ends to tie:

1. How do we use cryptography to find hard instances of NP-complete problems? For example, how do you sample a graph that's Hamiltonian, but for which a Hamiltonian cycle is hard to find? Or, how do you sample a graph such that it's hard to determine — non-negligibly better than by guessing randomly — whether or not it is Hamiltonian?

2. How do we patch up the trapdoor commitment scheme from class so that P^* does not learn how cheat by observing V open its commitment to a string of P^*'s choice?

Recall that every one-way permutation f has a hard-core bit B. Consider the following language:

$$L = \{y \mid \exists x \text{ such that } f(x) = y \wedge B(x) = 0\}$$

Suppose there exists a one-way permutation f over k-bit strings for every k.

a. Show that, there is no PPT algorithm that, on input a random y, determines whether $y \in L$ with probability non-negligibly better than 1/2.

b. Show that $L \leq_P \mathsf{HamCycle}$, where $\leq_P$ denotes polynomial-time reducibility.

c. Give a PPT algorithm that, on input the security parameter 1^k outputs a graph G such that no PPT A can determine whether G has a Hamiltonian cycle non-negligibly better than by random guessing.

d. Give a PPT algorithm that, on input the security parameter 1^k, outputs a graph G and its Hamiltonian cycle H such that no PPT algorithm can, on input G, find its Hamiltonian cycle with non-negligible probability.

e. Consider another language L_2, defined as follows:

$$L = \{\langle y_1, y_2\rangle \mid \exists x \text{ such that } (f(x) = y_1 \vee f(x) = y_2) \wedge B(x) = 0\}$$

Using techniques you developed in part (d), give a PPT algorithm that, on input the security parameter 1^k, outputs a graph G and its two distinct Hamiltonian cycles H_1 and H_2 such that no PPT algorithm can, on input G, find a Hamiltonian cycle of G with non-negligible probability.

f. Recall the almost final version of the Feige-Shamir protocol that we saw in class. It went as follows:

(a) Verifier computed and sent to the Prover a graph S for which it knew a Hamiltonian cycle T. S serves as the setup for the trapdoor commitment scheme (you will have to look up your notes from class or the FS paper to see how). V uses S to commit to a k-bit string; let this commitment be $(\alpha_1, \ldots, \alpha_k)$.

(b) Prover executes, k times in parallel, the first round of the Blum proof system, and sends to the Verifier a commitment to the resulting messages, using the trapdoor commitment scheme keyed by S. (Again, use your notes or the FS paper to remind yourself how that works.) The prover also sends to the Verifier a k-bit challenge $\beta_1, \ldots, \beta_k$.

(c) The verifier uses the trapdoor for S (i.e. its Ham cycle) to open his commitments to the value $\beta_1, \ldots, \beta_k$. The verifier also executes the second round of the Blum protocol.

(d) The prover checks that the Verifier opened his commitments correctly. If so, it executes the third round of the Blum protocol (k times in parallel).

(e) Finally, the Verifier runs the Blum verifier k times, and if it accepts every time, accepts.

In class, we saw that this proof system was black-box zero-knowledge (because the simulator could extract the trapdoor from the Verifier by resetting him to step 2). We ran into trouble with computational soundness: we needed a way to show that if G was not Hamiltonian, then the cheating prover P^* managed to compute the Hamiltonian cycle of S. Yet, in order to get to the end of the protocol with P^*, the reduction needs to already know the Ham cycle, so it runs into trouble!

Suppose that S is generated as in part (e). Note that either H_1 or H_2 from part (e) is a good enough trapdoor. Also note that the knowledge of H_1 does not help in finding H_2 (if you have done the reduction in (e) correctly). Show that, no matter which trapdoor V uses, the view P^* receives in the protocol we saw in class is the same.

g. In order to fix this protocol, show that, if G is not Hamiltonian, then our reduction, knowing either H_1 or H_2 (but not both) can use the cheating prover P^* to compute the other cycle (the one it does not know). Explain why this would contradict the fact that f is a one-way permutation.

Problem 3: Everything Provable is Provable in Zero-Knowledge

Recall that IP is the class of all languages that admits an interactive proof. In class, we saw a zero-knowledge protocol for the NP-complete language of Graph 3-Colorability which proves that all languages in NP have a zero-knowledge protocol. We will now try to prove a stronger statement which was proved by Ben-Or et al. [BOGG+88].

Any language that has an interactive protocol also has a zero-knowledge interactive protocol.

Let (P, V) be the interactive protocol for a language L. Towards constructing a zero-knowledge protocol, we will construct a protocol (P', V') which depends on (P, V).

a. As a first idea, let P' send messages of P in a committed form, instead of sending them in the clear. What are some immediate obstacles to this approach? Can you design a V' that can still work?

b. Useful fact: Any language that has an interactive protocol also has an AM protocol, where a verifier's messages consist only of truly random strings independent of the previous messages. Does this help in overcoming the obstacle that you identified in (a)?

c. Consider an AM protocol where P' sends messages of P to V' in a committed form. We will use P' to define a "useful" NP language. Can you think of a way to transform the verification step of V', where V' looks at the transcript of messages T of the protocol (P', V'), and transforms it to an instance T for an NP language L' such that, $T \in L'$ if and only if T is a transcript for an accepting (P, V) protocol.

d. Once we construct the NP language L', we can now use the fact that any language in NP has a zero-knowledge protocol. Can you describe the resulting zero-knowledge protocol (P', V')? How many rounds does the resulting zero-knowledge protocol have? Note that this will depend on the number of rounds of the interactive protocol.

e. Prove that the protocol is sound and zero-knowledge. Does soundness or zero-knowledge rely on the underlying commitment scheme having any properties? You may assume that the AM protocol has perfect completeness.

References

[BOGG+88] Michael Ben-Or, Oded Goldreich, Shafi Goldwasser, Johan Håstad, Joe Kilian, Silvio Micali, and Phillip Rogaway. Everything provable is provable in zero-knowledge. In *Conference on the Theory and Application of Cryptography*, pages 37–56. Springer, 1988.