Problem Set 5

This problem set is due on *Thursday, October 18, 2001* at the beginning of class. Late homeworks will *not* be accepted. You are to work on this problem set in groups of three or four people.

Mark the top of each sheet with your names, 6.857, the problem set number and question, and the date. **Type up your solutions** and be clear. Each problem should begin on a new sheet of paper. That is, you will turn in each problem on a separate pile of paper.

Remember to **cite** your sources of information.

Problem 5-1. Zero Knowledge

(a) I'm not going to tell

Let p be a prime and g a generator mod p. Given $y = g^x \bmod p$, the prover claims she knows the discrete logarithm x of y. p, g, y are public values. The prover wants to convince a verifier that she knows x without actually revealing x. She does so by the standard scheme of sending $u = g^r \bmod p$ to the verifier for a random r. The verifier sends back a challenge message, c. The prover responds with the message $v = r - cx$. The verifier then checks that $u = y^c g^v$.

Explain how this protocol has the properties of completeness, zero knowledge, and soundness. (See lecture notes 09 and definition 10.20 in handout 18.)

Draw a diagram showing the messages between the prover and the verifier. Explain the messages in your protocol and what role each fulfills.

(b) Signatures

Explain a modification to this scheme which enables the prover to sign a message and the verifier to verify the signature.

Hint: Let x be the prover's secret key and $y = g^x \bmod p$ and p be the public key. Transform your interactive ZK protocol in part (a) into a non-interactive ZK protocol as discussed in lecture.

(c) Equality of discrete logs

Unsatisfied with mere ZK proofs with a single discrete log, the prover wants to convince a verifier that she knows a discrete log x such that $y_1 = g^x \bmod p$ and $y_2 = h^x \bmod p$ where p is prime, but she does not want to reveal x to the verifier. g and h are generators of Z_p^*. The verifier knows p, y_1, y_2, g, h. The discrete log of h to the base g is unknown to anybody.

Construct an interactive zero knowledge protocol for this. Just like part (a), show how your protocol has the properties of completeness, zero knowledge, and soundness.

(d) Non-interactive version

Now create a non-interactive ZK version of your protocol from part (c).

Hint: use the same tactics from part (b).

(e) Blind ZK proofs

Now use the ZK system from parts (c) and (d) to create a blind signature scheme. In this system, there are users and an authority. Users will ask the authority to make blind signatures. Users can then publish the unblinded signature with the message via an anonymous channel.

Show how to use the schemes from parts (c) and (d) to implement a blind signature scheme.

Explain how the authority cannot learn anything about the message that the user wants signed.

Explain how the authority cannot link a message to a user. Make sure that a user can publish a proof that two values have the same discrete log without the user actually knowing the discrete log. After the authority sees many signed (unblinded) messages from the anonymous channel, it cannot link those signed messages to a particular user who previously asked for the blind signature. You can assume there are n users in the system and that unlinkability means the authority cannot link a message to a particular user with a probability greater than $\frac{1}{n}$.

Hint: The authority can use the transcripts of communication between a simulated prover and verifier to make blind signatures. The authority knows the discrete log x and hence can simulate the protocol. The authority could then send this transcript to the user as the blinded signature...

Problem 5-2. Scarfo case

In a recent court filing, the FBI described how they used key logging software in order to recover passwords of a suspected criminal. See `http://www.epic.org/crypto/scarfo/murch_aff.pdf` for more details.

Suppose that an adversary can see all of the keys that you type on your keyboard. Is it possible for you to securely type in a password to a program that is running on your machine? You can assume that the adversary can only log your keystrokes, and that you can make small modifications to the program which accepts your password. Either describe a method or argue why it is impossible. [Note, it would be best if you do not require a third-party device, such as a calculator, to complete logins.]