Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ron Rivest

Handout 6
April 22, 2014
Due: May 5, 2014

Problem Set 5

This problem set is due on *Monday, May 5* at **11:59 PM**. Please note that no late submissions will be accepted.

Please submit via MITx.

You can work on this problem set with a group of three or four students of your choosing. If you do not have a group, please email `6.857-tas@mit.edu` and we will assign you to a group. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration.

Homework must be submitted electronically! Each problem answer must appear on a separate page. Mark the top of each page with your group member names, the course number (6.857), the problem set number and question, and the date. We have provided templates for LaTeX and Microsoft Word on the course website (see the *Resources* page).

Grading: All problems are worth 10 points.

With the authors' permission, we will distribute our favorite solution to each problem as the "official" solution—this is your chance to become famous! If you do not wish for your homework to be used as an official solution, or if you wish that it only be used anonymously, please note this in your profile on the homework submission website.

Problem 4-1. Zero-knowledge proofs

Given a set of elements $S = \{1, 2, \ldots, n\}$ and k subsets $S_1, S_2, \ldots, S_k \subseteq S$ such that union of all S_i equals S, the set cover problem is to identify the smallest subset of $\{S_1, S_2, \ldots, S_k\}$, whose union equals S.

In this problem we are asking for a zero-knowledge protocol for the set cover problem. That is, devise a way for a prover P to convince a verifier V that she knows a set cover of size at most ℓ: $S_{i_1} \cup S_{i_2} \cup \ldots \cup S_{i_l} = S$. Your protocol should not reveal information about the chosen cover that the verifier could not compute himself.

Physical protocols (like ones for sudoku we discussed in class or card-based examples we saw) are also acceptable.

Problem 4-2. Digital signatures

When you send an email, it is transmitted through many servers, each of which can potentially modify your message. Luckily we can use public key cryptography to sign our messages, and thereby allow others to verify their integrity. We can also use PKI (like OpenPGP) to safely distribute our public keys.

Figure out how to send a digitally signed message using your current mail client to your other project team members. Verify the digitally signed messages received from your project team members. If your current mail client doesn't support signatures, you can download and use Thunderbird with the Enigmail extension.

(a) Write up the steps you needed to do the above. (Include a description of your mail client, etc.) What certificates did you have to work with?

(b) Have **each member** of your team send a digitally signed message to `6857-staff`. (Note: the staff needs to be able to verify your signature for you to get credit for this problem! You may need to get a certificate to them somehow. We suggest posting your certificates to a PGP keyserver such as `pgp.mit.edu`.)