
Problem Set 5

This problem set is due *online*, at <https://sec.csail.mit.edu/> on *Wednesday, April 21* by 11:59pm. No late submissions will be accepted.

You are to work on this problem set with a group of 3 students of your own choosing. We also allow you to work with your final project team, even if it doesn't have exactly 3 students. 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 L^AT_EX and Microsoft Word on the course website (see the *Resources* page).

Grading: Each problem is worth 20 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 3-1. Digitally Signed Email

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 6.857-tas. (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.)

Problem 3-2. Encrypted File Systems. A typical encrypted filesystem is all-or-nothing: either a user knows the secret key and has access to the entire filesystem, or the user doesn't know the key and the filesystem is entirely opaque to him. In this problem, you will design an encrypted filesystem having more flexibility.

On top of the standard Unix permission model, we introduce a concept called *admittance*, which you should implement cryptographically in your filesystem. Here is the security policy we would like your filesystem to implement:

- A user who is admitted to a *file* has exactly those privileges that are specified by the standard Unix permission flags for that file.
- A user who is admitted to a *directory* has exactly those privileges that are specified by the standard Unix permission flags for that directory, *and* is admitted to all files and sub-directories of that directory.
- A user who is admitted to a file or directory may grant admission to that file or directory to any other user.
- If a user is not admitted to a file or directory, that file or directory (even its existence) should remain completely hidden to the user. That is, the user must not learn any information about file names, file sizes, modification times, contents, etc. of the file or directory.

- (a) Design an encrypted filesystem that implements the above policy as well as you can. Clearly state any cryptographic tools that you use and assumptions you make. You will probably want to refer to documentation on the basic design of Unix filesystems (cite any references you use). Don't worry about supporting symbolic/hard links or other advanced filesystem features.
- (b) Suppose a malicious user has the ability to read all the blocks on the disk containing your encrypted filesystem. The malicious user may also have legitimate admission to certain files and directories. What additional information about the filesystem might be leaked by your design, given complete access to the disk contents?
- (c) Suppose we want to add an additional feature to the security policy, which is: the owner of a file/directory may *revoke* admittance of that file/directory from another user. Discuss some of the significant difficulties that arise in your design when trying to implement this new policy. (You need not fix these difficulties.)