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

Handout 10
April 26, 2011
Due: May 6, 2011

Problem Set 5

This problem set is due *via email* to `6857-tas@mit.edu`. By Institute regulations, the problem set is due Friday, May 6, at 11:59 PM, but may be turned in without penalty when you turn in your final project report.

You are to work on this problem set in your project groups. 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 and Late Policy: Each problem is worth 10 points. Late homework will not be accepted without prior approval.

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 on your homework.

Problem 5-1. Buffer Overflows

We've set up a virtual machine at `hulciber.mit.edu` with username/password `guest`/`guest`. There's a `ta` account on the server with some privileged files in `/home/ta/files`. These files are not world-readable, but there's a setuid-`ta` program in `/home/ta/hackme` to let you read them. (That is to say, when anyone runs that program, the program runs with effective user id `ta` and therefore has access to `ta`'s files.) The source is also in that directory as `/home/ta/hackme.c`.

The program is vulnerable to a buffer overflow. First, the same program can be used by the `ta` account to *write* to the files; if you can trick the program into believing you're running as that account, you can edit the files too.

(a) These days, GCC has built-in protection against buffer overflows. One of its security measures is to reorder the variables on the stack to place buffers after other local variables. Explain what effect this measure would have on `hackme`.

(For the purposes of this problem, we've compiled the code with the `-fno-stack-protector` flag, which disables this measure as well as a few others.)

(b) Execute a buffer overflow attack to add some text of your choice to a file. Include the string you used in your solution.

(c) Explain how to craft an input string that would execute a shell, therefore allowing you to run arbitrary commands as the `ta` account.

(d) (optional) Execute the attack from part (c). Prove that you managed to get a shell by creating another file in the `files` directory. Include any code used in the attack.

We've installed GCC and GDB on this machine, turned off ASLR, and marked `hackme` to have an executable stack. If you'd like other software installed, or think you need some other security setting changed, let us know.

Problem 5-2. 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 6857-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.)