# Homework 5

**Total Points:** 200 **Due Date:** Friday, December 11, 1998

1. (*100 points*) Please write a library function called *my_system*. The function *my_system* provides a safe environment for the execution of the given command. Specifically, it first deletes the caller's environment, and sets only the following environment variables: **PATH** (to "/bin:/usr/bin:/usr/ucb") **IFS** (to " \t\n"), **SHELL** (to the current value of the **SHELL** environment variable, or to the Bourne shell if *SHELL* is not set) and **TZ** (to "PST8PDT"). It sets umask to 077 and then spawns the shell named in **SHELL** to execute the given command.

   The interface to *my_system* is identical to that of *system*(3), as is its functionality.

   ***Hints*** :To write the function, in the parent ignore all signals. Then *fork*, and restore the previous signals in the child. (Use *signal*(3) for this.) Clean up the environment in the child, and then *execve*(2) the desired program. Meanwhile, in the parent, we do a *wait*(2) for the child process, and when the child terminates, we restore the old values of the signals. Please change the process environment after the *fork*(2). if you change it before the *fork*, you will change the parent process' environment, which is not correct!

   You may find the program *printenv*(1) useful; it prints a list of all environment variables defined to it, so you can use it to see what environment variables are defined

   ***Note*** : you may use any language you like, but the result must be a library function that can be called from a C program.

   Also, the program ~cs153/tester.c is a primitive driver that you can use to test your library routine. (We will use the same sort of program, but with canned commands, to test your library function.) You need to create a *Makefile* that will compile your library routine, the tester.c program, link them, and call the executable *tester*. Please submit this makefile, along with your library routine. **Do not submit the tester.c program** because we will use a modified version of the sample!

2. (*100 points*) Please read the book *Rules for Radicals* by Saul Alinsky. Pick any two of the principles discussed in the book (either explicitly or implicitly; in the latter case, show how you derived the principle in a paragraph or two), and compare and contrast them with the way attackers would use Trojan horses, computer viruses, and other examples of malicious code to compromise the security of a system. Your essay should be at least two pages (120 lines) long.

## Extra credit

3. (*10 points each)* Implement a set of library functions to enable the caller to control the environment set in *my_system*:
   a. *int le_set*(*char *env*) takes an environment variable and associated value in the form "VAR=value". Subsequent calls to *my_system* will add that environment variable and value to the subprocess. If *env* is "VAR=", the variable is defined to have the empty value. If *env* is "VAR", the value from the current environment is used. This function returns 1 if the argument was successfully added, and 0 if not.
   b. *int le_unset*(*char *env*) deletes an environment variable from the environment of subsequent calls to *my_system*. The argument *env* names the variable, so to delet "VAR", say "le_unset("VAR")". This function returns 1 if the argument was successfully deleted, and 0 if not.
   c. *void le_clobber*(*void*) forces the next call to *my_system* to use the standard *my_system* environment.
   d. *int le_umask*(*int umask*) changes the value of *umask* as indicated in the environment of subsequent calls to *my_system*. This function returns 1 if the change succeeds, and 0 if not.