

Lab #1

Due date & Time 11:59pm on September 10, 2007. Hand in the report to TA during PSO, or email to the TA (zmao@purdue.edu) by the due time.

Late Policy: Late labs will not be accepted.

Additional Instructions: The submitted report must be typed. Using Latex is recommended, but not required.

Solaris has two access control mechanisms to protect files and directories. One is the 12 permission bits associated with each file. The other is a more complicated ACL mechanism accessed via the `setfacl` and `getfacl` commands. Rights amplification is available with the `setuid` and `setgid` commands. In this lab, you are asked to play with those two mechanisms based on the instructions given below.

Go through “<http://www.hccfl.edu/pollock/AUnix1/FilePermissions.htm>” before the lab. Read the man pages for related commands and experiment with them. Do each of the following and “turnin” output (from the `ls` command, running script, or other proof) that show what you did. Get someone else in the class (a buddy or two) to also turnin output to show that what you did works.

1. (10 pts) Create a directory with files in it and use the `chmod` command to set the permission bits so that anyone on the system except you can remove the files, but only you can modify the files.
2. (10 pts) Create a directory with files in it and use the `chmod` command on the directory only so that you can list the file names but not open or delete any of the files. Also, set the permissions on the directory and files so that anyone else on the system can open the files but cannot list their names, that is, they can access the files if they know the names, but they cannot find out the names with `ls`.
3. (15 pts) Recall that a directory under Solaris is just a special file with names and pointers to files (inodes). Does this fact help to explain the behavior of the permission bits you observed above? Explain.
4. (15 pts) Explain why do we need the `set-uid` bit for executables. You can explain by using a specific example, e.g., the program “`login`” or “`passwd`”.
5. (30 pts) Write a C or C++ program that will `chdir()` into a directory you have created with `chmod` mode 700 and then into a subdirectory with mode 777. Then it will set its effective UID to its real UID, and create a sentinel file before exiting. Use `chmod` to make the executable program `setuid` to your account and have someone else run it. (Setuid means that the program will start with its effective UID equal to the owner of the file and not to the user account running the program.) What happens? Can someone other than the superuser create the sentinel file without using your program?

6. (10 pts) Create a default ACL on a directory so that any new files created in it have read/write access by you and by user zmao, have read access by user li83, and have no access rights by any other user. Create some files to demonstrate this behavior.
7. (10 pts) In the directory you just created with the default ACL, put a specific ACL on a file to give user ninghui read-only access, user zmao write-only access, yourself no access, and everyone else read/write access. Show that this does not disturb the default ACL on the other files.