Homework 4

Total Points: 250 **Due Date:** Monday, November 27, 1998

1. (*100 points*) Please write a program to determine if a named user can access a named file or directory. The program is to be called *access*, and the command line looks like:

   ```
   access options user file
   ```

   Here, *user* is either a user's login name or UID, *file* is a file (or directory, or other entity), and *options* is any combination of **r** (for read), **w** (for write), and **x** (for execute), with a leading **–**. If none of **r**, **w**, or **x** is given, act as though all three had been given. Your output should say:

   ```
   user can read file
   ```

   or

   ```
   user can read and write file
   ```

   and so forth. If the option **l** is given, then every component of the path is to be checked as well as the file itself.

   If the *file* cannot be accessed, any command line argument is malformed, or an error occurs in your code, print an appropriate error message and exit with an exit code of 1. Otherwise, exit with an error code of 0. Please write your code robustly, and comment it!

   Example . The output of

   ```
   access -rwx bishop /home/cs153/public_html
   ```

   should be

   ```
   bishop can read and execute /home/cs153/public_html
   ```

   The output of

   ```
   access -l bishop /home/cs153/public_html/index.html
   ```

   should be

   ```
   bishop can read and execute /
   bishop can read and execute /home
   bishop can read and execute /home/cs153
   bishop can read and execute /home/cs153/public_html
   bishop can read /home/cs153/public_html/index.html
   ```

   Submitting your programs . If you write your programs in a compilable language (like C), you ***must*** submit a makefile to compile your source code. A single *make* should generate the *access* executable. ***Do not*** submit *access* executables unless it is written in a scripted language (such as shell script or Perl). Your program should run correctly on any of the CSIF machines (that is, on the DECs, the HPs, and the SGIs).

2. (*20 points*) Can the Trojan Horse attack work on a system protected by capabilities?

3. (*10 points*) Some early computers protected their operating systems against destruction by placing it in memory locations from which all programs (including the operating system itself) could read, but into which no program could write. What problem does this protection scheme create?

4. (*20 points*) Prove or disprove: the power set of any set and the relation "subset" form a lattice. To prove this, show how to construct a greatest lower bound and a least upper bound (under "subset") for any pair of elements of the power set, and prove that "subset" is transitive, associative, and antisymmetric. To disprove this, exhibit a specific counterexample to any of these properties.

5. (*100 points*) Please read the book *The Shockwave Rider* by John Brunner, and contrast his use of the terms "worm," "virus," and other references to malicious logic with the way we use those terms now. As with previous essays, please write well. Your essay should be at least two pages (120 lines) long.

Extra credit

6. (*20 points)* Why do most UNIX systems limit the use of the *chroot*(2) system call to the super-user? (Hint: think about protection problems.)

7. (*30 points*) Determine which of the systems in the CSIF provide login/audit UIDs and GIDs, and saqved UIDs and GIDs. Write a program to print each of these, as well as the real and effective UIDs and GIDs.