Homework 1 (50 points)

cs3157 – Advanced Programming
Prof. Shlomo Hershkop
Dept of Computer Science
Columbia University
Summer 2006

Out: May 25.
Due: June 9 1pm

Objective:

1. Practice with some Perl coding/compiling/debugging
2. Get a taste of what Security programmers do.
3. Have fun!

Programming Part:

There is a tool provided as part of the standard Linux install which is called "locate". This program allows you to search across your files for a specific file or directory. It doesn't actually do a real time search, but searches an internal file containing information about your file system. The command to trigger a refresh of this information is called "updatedb". (you can try it if you have cygwin installed, but it might take a while!).

In this homework you will be building a "super" locate command. It will be part locate and part security tool. This tool can not only locate a specific file, but can also give you a unique signature to identify the file….when you run it over your files, it is called taking a snapshot.

Let's imagine someone has hacked into your computer and manipulated some set of files!!! That is very bad! (I was teaching a class at the time, so you can't pin this on your instructor).

Lucky for you, you have a snapshot of your system in which you noted each file, and their unique signature. This means you can easily take a new snapshot and run through all your current files and compare them to see which have been changed.

If the signature is not the same, it is an indication that something was changed. (Note we can manipulate a file without changing its size, so size is not an indication).

In addition to "just" detecting file changes between two snapshots, we would like to use this tool to take a snapshot and then be able to search for a specific file to get any individual signature. When searching you should be doing pattern matching not just straight equals. So if you search for "time" it should match "Time", "timestamp", "stoptime" etc.

So you will need to create the following Perl program:

1) Name it **`superlocate.pl`** and program it in strict mode. (1 POINTS)

2) You will be judged on comments and formatting too! So please remember to document what you are doing and clean up your code with an automatic formatter or good practice. (6 POINTS)

3) It should work in the following manner:
    a. When it starts it should present the following choices:
        i. Create Snapshot
        ii. Browse Snapshots
        iii. Find File
        iv. Compare 2 snapshots
        v. About
        vi. Quit

    b. Create Snapshot (12 points)

        will allow the user to create a snapshot by allowing the user to enter a ***PATH***. It will recursively run through the location specified by ***PATH.*** For each **file** you will need to take it's MD5 hash (of contents), and keep track of it in the snapshot file.
        for example if the user enters c:\temp
        it will start at c:\temp and go down to all files there, and put the output into a file in a **snapshot** subdirectory. The name of the file should be also specified by the user.

        You have to keep a relative path when keeping track of the files.

    c. Browse snapshot (7 Points)

        should allow you to list all the files in the snapshot directory and see more info (like when it was created).

    d. Find File (10 Points)

        Will ask for a name and allow you to specify one of the current snapshot files to search

e. Compare 2 Snapshots (10 Points)

   will allow you to specify 2 snapshot files and output 3 lists:
   i. **Changed File**s: those are files which appear in both lists but have different MD5s
   ii. **Deleted Files**: those are files only appearing in the first list
   iii. **Created Files**: those are files only appearing in second list

f. About (2 Point)

   Short message about your program

g. Quit (2 Point)

   will quit your perl script

Extra Credit (5 Points): Allow you to pass in all arguments for Create, Find, or Compare so can run this tool without interaction….

TIPS:
In addition to submitting the Perl program, you will need to include a README file giving a sentence of what each file in the submission does.

Your code should never terminate unexpectedly. Check for all possible errors the user could make (non-existent file, non-existent directory, etc). When any user error occurs, exit gracefully with an appropriate error message

Make sure to divide your program into logical sections using subroutines and not one giant main as discussed in class! Also remember to `use strict;`

Code Style is graded on how easily read your source code is. For generic tips on improving your code's style, read **perldoc perlstyle**. Most important are three criteria: meaningful variable names, consistent and helpful indentation, and explanitory but not over-abundant comments.

Start early and Good Luck!!