Secure Scripting

**The Basics**

# Unit 1 lab Solutions

### Lab Exercise 1

Find the words in the word list “dict.txt” that contain the string “gry”.

1. What is the format of the file “dict.txt”? How many words per line does it contain?

The file “dict.txt” contains a list of words, one word per line.

1. Look up the program *grep*. What arguments would you give it to look for the string “gry” in dict.txt?

The command is “grep gry dict.txt”, which prints each line containing the string “gry”.

1. Write a two-line shell script called “lookfor1.sh” that executes this command to look for words containing “gry” in dict.txt. The first line should specify that the program “/bin/sh” is to be used. The second line should contain your command. Try it out. If you did it right, you will see eight words, the first being “agrypnia” and the last being “pouggry”.

See the script lookfor1.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

### Lab Exercise 2

Find the words in the word list “dict.txt” that contain a string supplied by the user. Your shell script is to be called “lookfor2.sh”.

Hint: Begin with your script from Lab Exercise 1 and modify it as indicated.

1. How do you represent the first argument from the command line in the command you put into your script?

$1

1. Modify the script you wrote for Lab Exercise 1 to take the string to be searched for from the command line. Test your script by searching for the strings “hello” and “world”.

See the script lookfor2.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles. When you give the argument “hello”, you should find four words. When you give the argument “world”, you should find thirty-eight words.

1. What happens if no arguments are given? Two arguments?

If no arguments are given, you get an error. If multiple arguments are given, the first is treated as the string and the rest as filenames.

1. How would you embed a blank character in your argument?

You would surround the argument by single quotes (double quotes also work), so to look for the string “hello world” you would type:

lookfor2a.sh “hello world”

1. What happens if an argument contains a blank?

If the argument contains a blank, it will normally be read as two separate arguments. The problem is that the full argument, with the embedded blank, is passed to the script as a single argument. When the substitution for $1 occurs, it looks like two words, not one.

1. Modify your script so that an argument with a blank is handled properly (that is, searched for in the file dict.txt).

Add quotation marks around the $1. The quotation marks must be double quotes; single quotes won’t work. See the script lookfor2a.sh in 06.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

### Lab Exercise 3

Modify the script you wrote for Lab Exercise 2 by adding an “if” statement that checks whether there is exactly one argument.

If there is *not* exactly one argument, your script should print the error message “Usage: give exactly 1 argument, the string to be looked for” and exit immediately.

See the script lookfor3.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

### Lab Exercise 4

Find the words in the word list named by the user (like “dict.txt”) that contain a string supplied by the user. Your shell script is to be called “lookfor4.sh”.

Hint: Begin with your script from Lab Exercise 2 and modify it as indicated.

1. Modify your script in Exercise 2 so that the word list is the second argument on the command line. Remember to change your “if” statement so it balks if there are not exactly two arguments!

See the script lookfor4.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

1. Change the error message to “Usage: lookfor4 string file”.

See the script lookfor4a.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

1. Change the error message so it prints the exact name of the script. That is, if you call the script “find4” and not “lookfor4.sh”, the error message should print as “Usage: find4 string file”. A similar error message should occur if you call the script “catdog” *without changing the script!*

See the script lookfor4b.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

### Puzzler

In the script you wrote for Lab Exercise 4, if the file does not exist, *grep* prints an error message. This will be confusing to naïve users. Add a test at the beginning of that script that prints the error message

*script\_name*: file *file\_name* cannot be read

where *script\_name* is the name of the script and *file\_name* is the name of the file that the user gives.

See the script lookforp1.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

### Big Puzzler

This is for all you Linux experts. It demonstrates that there are several ways to write a script.

1. Rewrite the script you wrote in Lab Exercise 4 using the program *sed* rather than *grep*.

See the script lookforp2.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.

1. Rewrite the script you wrote in Lab Exercise 4 using the program *awk* rather than *grep*.

See the script lookforp3.sh in 09.SeS\_Unit1\_TheBasics\_LabSolutionFiles.