SECURE SCRIPTING (PYTHON)

**ADVANCED CONTROL**

# unit 2 lab solutions

### Lab exercise 1

This exercise starts you off. You do not need to write any scripts until Part D. First, list the attributes of the files in the directory “sample”, one per line.

1. When you execute the command *dir* with the */n* option, which of the desired attributes are printed?

Last date and time the file was modified, type(directory or file), size(if file), and the name of the file or directory.

1. What does the command *certutil -hashfile* print?

SHA1 hash of *filename*: (hash of the file)

1. How would you cause both outputs to be printed on the same line?

output1 = check\_output(["dir", "/n", fi])

output2 = check\_output(["certutil -hashfile",fi])

print (output1,output2)

where fi is the variable containing the filename.

This is just one of many ways to do this.

1. Write a shell script that uses a *for* loop to list the attributes of the files in the directory “sample”, one per line.

See the script scan1.py in Module 2 - Lab Scripts.

1. Lab Exercise 2

You are now going to modify the script you write for Lab Exercise 1D to see if a master file exists and, if it doesn’t, create one. Then, in the loop, you will ignore any non-regular files and the master file.

1. At the beginning of the script, add a line that defines a variable called MASTER, and give it the initial value “MasterList”. When you are done, save a copy because you will use this in Lab Exercise 3.

See the script scan2a.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. After the variable definition but before your loop, test to see if the file named by the value of the variable MASTER exists. If so, print the message that the file “exists; please delete it” and exit, giving exit status code 1. If not, create it.

See the script scan2b.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. In the *for* loop, before you print the file attributes, check to see if the file being examined is a regular file or the MasterList file, and if not, immediately go to the next file (that is, do not get the file attributes).

See the script scan2c.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. Modify your script so that the attributes are stored in the file named by the value of the variable MASTER.

See the script scan2d.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. Finally, if everything works, have the script exit with an exit status code of 0.

See the script scan2e.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

### Lab Exercise 3

This exercise uses the copy of the script you saved after completing Lab Exercise 2A. We will be modifying it in a way similar to the rest of Lab Exercise 2.

1. At the beginning of the script, add a line that defines a variable called TMP, and give it the initial value “tmppid”. (pid is the process id extracted from the system) Then add another line that creates the file named by the value of TMP. What is the actual name of the file created?

See the script scan3a.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. In the *for* loop, before you print the file attributes, check to see if the file being examined is the file named by the value of TMP, and if so, *immediately* go to the next file (that is, do not get the file attributes).

See the script scan3b.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. Modify your script so that the attributes are stored in the file named by the value of the variable TMP.

See the script scan3c.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. After the loop, print the message “Changed files:”, and then compare the contents of the files named by TMP and MASTER. Use *fc.exe* to generate the comparison. What happens if the file named by the value of MASTER does not exist? Note that hidden files and empty files will not be able to be hashed.

See the script scan3d.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts. If the master file does not exist, you get an error message.

1. Before the loop, check that the master file (the file named by the value of MASTER) exists. If it does not, print an error message saying “Master file does not exist; please generate it” and exit with an exit status code of 1.

See the script scan3e.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. Conclude by using the command *del* to delete the file named by the value of TMP after the comparison in Part D. Again, have the script exit with an exit status code of 0.

See the script scan3f.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

When you test this script, the MasterList file must already exist. If it does not, execute the script you wrote in Lab Exercise 2 (scan2e).

You should get at least one line for the file MasterList. You will also get other lines corresponding to the scripts you wrote. As long as you get the line for MasterList, you’re doing it right.

### Lab Exercise 4

Now combine the scripts you wrote for Lab Exercises 2 and 3. Your script should define the variables TMP and MASTER, then do the parts of Lab Exercises 2 and 3 in the following order:

2A, 3A, 3B, 2C, 3C, 3D, 3E, 2E

Do not include Exercises 2B or 2D in this script. Then, before the line you wrote for Exercise 3D, have the script print “Changed files:” and add the following on the same line as the command you wrote for Exercise 3D:

proc3 = subprocess.Popen(["fc", MASTER, TMP],stdout = subprocess.PIPE, shell=True)  
(fc, err) = proc3.communicate()  
print (fc.rstrip().decode("utf-8"))

See the script scan4.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

Run your script twice to test it. First, just run it. The list of changed files should have only the ones you have written and left in the directory. Then change the time of last modification of the file “abcde” by going into the abcde file, typing something, deleting what you typed, and saving it.

Now rerun the script. The list should be the same as before but with the addition of a change in “abcde”.

### Lab Exercise 5

Now you will modify the script from Lab Exercise 4 to handle two options. First we will handle the option –g, which creates the master file, then -d, which deletes the master file. If the script is called with no arguments, it will generate a list of files the attributes or contents of which have changed since the master list was created.

For the –g option:

1. Create a variable called GENMASTER and, at the beginning of the script, set it to “no”.

See the script scan5a.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. If the –g option is given, set GENMASTER to “yes”. What happens if you give some other argument or option, like “-m”?

See the script scan5b.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts. If you give any option other the –m, it ignores the option; indeed, it does not even give an error message, which means it has a terrible user interface. The next part fixes that.

1. After the argument processing loop, check to see whether GENMASTER is “yes”. If it is, do what you do in the script that was the answer to Lab Exercise 2; you can just copy it into this script if you like, but if you do, don’t copy the line setting the variable MASTER. Then exit with an exit status code of 0. If it is “no”, do what you do in the script that was the answer to Lab Exercise 3; again, you can just copy it into this script if you like, but don’t copy the lines setting the variables MASTER and TMP. Exit with an exit status code of 0.

See the script scan5c.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

Test your script by running it in the sample directory. First, do not give the –g option; you should get a list of files that have changed, most likely including abcde (from Lab Exercise 4). Then run it again giving the –g option. You should get the error message saying the master file exists, please delete it.

For the –d option:

1. Create a variable called DELMASTER and, at the beginning of the script, set it to “no”.

See the script scan5d.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. If the –d option is given, set DELMASTER to “yes”. If any command-line option (or argument) other than –d or –g is given, print the error message “Unknown option” followed by the option, and exit with an exit status code of 1.

See the script scan5e.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

1. Before you check the value of the variable GENMASTER, if the value of DELMASTER is “yes”, check that the master file exists. If it does, delete the master file and exit with an exit status code of 0. If it does not, print “Master file does not exist; please generate it” and exit with a status code of 1. Otherwise, if the –d option is not given, continue.

See the script scan5f.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.

Test your script by running it in the parent directory of the sample directory. First, do not give the –d option; you should get a list of files that have changed, most likely including abcde (from Lab Exercise 4). Then run it again giving the –d option. You should get the error message saying the master file exists, please delete it.

Finally, do some sanity checking. It makes no sense to give both the –d and –g options, so we need to give an error message if both are set.

1. Right after you process the arguments (options), check the values of DELMASTER and GENMASTER. If both DELMASTER and GENMASTER are “yes”, print the error message “Only one of –d, -g allowed” and exit with an exit status code of 1.

See the script scan5g.py in 10.SeSPython\_Unit2\_AdvancedControl\_LabScripts.