ELEC 377 Lab 4: Shell Scripting – Testing

Section 003, Group 16 – Thursday Lab

Erhowvosere Otubu, #20293052

Ivan Samardzic, #20296563

Date of Submission: November 14, 2023

Testing

Case 1 – No Flags Specified

In test 1 the program was executed with the following command, indicating that test 1 is running, and the output is redirected to file “t1lab.txt”. This is shown below:

```
./ps.sh > t1lab.txt
```

Visibly, this command is executed with no additional flags in the input. Therefore, the resulting output should only highlight the process ID (PID), and the name of the user (UID).

To test the accuracy of this execution, the standard “ps” command is executed to display the PID and UID in a redirected file called “t1system.txt”, as follows:

```
ps -eo pid,user > t1system.txt
```

With both our script, and the standard “ps” command now written to external files, the two can be compared to test the accuracy of our script. It is concluded that the resulting output is identical to that of the system’s “ps” command, and our script prints all running processes as expected, with only the PID and User columns.

The only outliers within the comparison are the few different process ID’s, which can be accounted for. The script ps.sh iterates through the /proc directory, and lists processes found within. This may not include the same system processes that are found when executing Linux’s “ps” command. The script may not have access to all system processes, and therefore the last few PID’s will never be identical. This same ideology is carried onto all of the test cases, but is still proven to be accurate.

Case 2 – Group Flag Specified

In test 2 the program was executed with the following command, indicating that test 2 is running, and the output is redirected to file “t2lab.txt”. This is shown below:

```
./ps.sh -group> t2lab.txt
```

Visibly, this command is executed with an additional “-group” flag in the input. Therefore, the resulting output should highlight the process ID (PID), the name of the user (UID), and the name of the group (GID).

To test the accuracy of this execution, the standard “ps” command is executed to display the PID, UID, and GID in a redirected file called “t2system.txt”, as follows:

```
ps -eo pid,user,group > t2system.txt
```

With both our script, and the standard “ps” command now written to external files, the two can be compared to test the accuracy of our script. It is concluded that the resulting output is identical to that of the system’s “ps” command, and our script prints all running processes as expected, with only the PID, User, and Group columns.

The only outliers within the comparison are the few different process ID’s, which have been explained in test 1, above. Our script may not have access to all system processes, and therefore the last few PID’s will never be identical.

Case 3 – Group and RSS Flags Specified

In test 3 the program was executed with the following command, indicating that test 3 is running, and the output is redirected to file “t3lab.txt”. This is shown below:

```
./ps.sh -group -rss> t3lab.txt
```

Visibly, this command is executed with additional “-group” and “-rss” flags in the input. Therefore, the resulting output should highlight the process ID (PID), the name of the user (UID), the name of the group (GID), and the RSS values in kB.

To test the accuracy of this execution, the standard “ps” command is executed to display the PID, UID, GID, and RSS in a redirected file called “t3system.txt”, as follows:

```
ps -eo pid,user,group,rss > t3system.txt
```

With both our script, and the standard “ps” command now written to external files, the two can be compared to test the accuracy of our script. It is concluded that the resulting output is identical to that of the system’s “ps” command, and our script prints all running processes as expected, with only the PID, User, Group, and RSS columns.

The only outliers within the comparison are the few different process ID’s, which have been explained in test 1. Our script may not have access to all system processes, and therefore the last few PID’s will never be identical. This also results in different RSS values for those respective processes.

Case 4 – Group, RSS, and Command Flags Specified

In test 4 the program was executed with the following command, indicating that test 4 is running, and the output is redirected to file “t4lab.txt”. This is shown below:

```
./ps.sh -group -rss -comm> t4lab.txt
```

OR

```
./ps.sh -group -rss -command> t4lab.txt
```

Visibly, this command is executed with additional “-group”, “-rss”, and either “comm” or “command” flags in the input. Therefore, the resulting output should highlight the process ID (PID), the name of the user (UID), the name of the group (GID), the RSS values in kB, and the name of the command (CMD).

To test the accuracy of this execution, the standard “ps” command is executed to display the PID, UID, GID, RSS, and CMD in a redirected file called “t4system.txt”, as follows:

```
ps -eo pid,user,group,rss,comm > t4system.txt
```

OR

```
ps -eo pid,user,group,rss,command > t4system.txt
```

With both our script, and the standard “ps” command now written to external files, the two can be compared to test the accuracy of our script. It is concluded that the resulting output is identical to that of the system’s “ps” command, and our script prints all running processes as expected, with all of the PID, User, Group, RSS, and Command columns.

The only outliers within the comparison are the few different process ID’s, which have been explained in test 1. Our script may not have access to all system processes, and therefore the last few PID’s will never be identical. This also results in different RSS values for those respective processes.

Case 5 – Comm and Command Flags Specified

The final test case is used to highlight the error occurring when both “-comm” and “-command” flags are specified. Specifically, the program was executed with the following command, indicating that test 5 is running, and the output is redirected to file “t5lab.txt”. This is shown below:

```
./ps.sh -comm -command> t5lab.txt
```

As expected, an error message is written since only one of comm/command is allowed to be printed.