Operating System (Lab)

Lab Task: 06

Submitted to
Mr. Meesum Raza

Submitted By
Syed Qasim Ali
(BSAI-161)

Junaid Asif
(BSAI-144)

Adeel Naeem
(BSAI-146)

Submission Date: November 25th, 2024

Question: 01

a. **Type in the example from the slide to display a list of users logged in. (Try running just the who command first to see what it displays.)**
b. **Arrange the list of usernames from the who command's output to be sorted and remove any duplicates.**
c. **Use the last command to display a record of login sessions, and then reverse it using tac. Which is more useful? What happens if you pipe the output into less?**
d. **Use the sed command to correct the misspelling of "enviroment" to "environment". Test this on a file containing a few lines of text. Does it work if the misspelling occurs multiple times on the same line?**
e. **Use the nl command to number the lines in the output from the previous question.**

Answer:

a. **Run the who command on its own:**

This command will list all logged-in users, including:

- The username.
- The terminal they are logged into.
- The date and time of login.
- The host from which they logged in (if applicable).
- The Who -h command adds a header to the output.

b. **Sorted usernames without duplicates:**

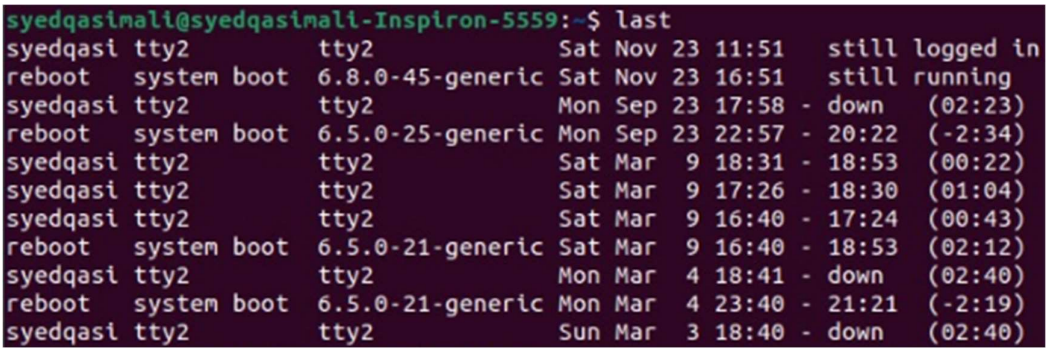
To sort the list of usernames in who's output and remove duplicates, you can use a combination of who, awk, sort, and uniq. Here's how you can do it:

- **who:** Lists all logged-in users.
- **awk '{print $1}'**: Extracts the first column, which contains the usernames.
- **sort:** Sorts the usernames alphabetically.
- **uniq:** Removes any duplicate usernames from the sorted list.

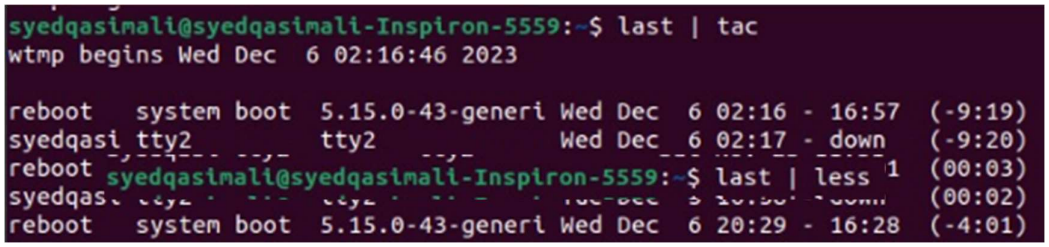
```
syedqasimali@syedqasimali-Inspiron-5559:~$ who
syedqasimali tty2         2024-11-23 11:51 (tty2)
syedqasimali@syedqasimali-Inspiron-5559:~$ who -H
NAME     LINE         TIME             COMMENT
syedqasimali tty2         2024-11-23 11:51 (tty2)
syedqasimali@syedqasimali-Inspiron-5559:~$ who | awk '{print $1}' | sort | uniq
syedqasimali
```

c. **Last commands:**

- **Last:** The last command displays a record of login sessions. It shows details like usernames, terminals, IPs (or hostname), and login/logout times.



- **Last | tac:** Using tac reverses the order of the last output, showing the oldest entries first.



- **Last | less:** Piping the output into less is especially useful if there is a lot of data. You can scroll through the output one screen at a time, it supports search functionality (e.g., type /user1 to find "user1") and also allows you to review long lists without overwhelming the terminal.

d. Sed command:

- Create a file named testfile.txt with a few lines containing the misspelling "enviroment".
- Run the **sed** command to replace all instances of "enviroment" with "environment".
- **s/enviroment/environment/:** Substitutes the first occurrence of "enviroment" with "environment".
- **/g:** The g flag ensures all occurrences on each line are replaced, not just the first one.
- The **sed** substitution with the **g** flag effectively corrects all occurrences of the misspelling, even if it appears multiple times on the same line.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ echo -e "The enviroment is changing rapidly. Protect the enviroment for future generations." > testfile1.tx
t
syedqasimali@syedqasimali-Inspiron-5559:~$ ls
Android  Documents  Public          StudioProjects  Videos
c++      Downloads  PycharmProjects  Templates
Desktop  Music      python          testfile1.txt
dev      Pictures   snap            testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ cat testfile1.txt
The enviroment is changing rapidly. Protect the enviroment for future generations.
syedqasimali@syedqasimali-Inspiron-5559:~$ sed 's/enviroment/environment/g' testfile1.txt
The environment is changing rapidly. Protect the environment for future generations.
syedqasimali@syedqasimali-Inspiron-5559:~$ echo -e "The enviroment is changing rapidly.\nProtect the enviroment for future generations.\nEnviroment aw
areness is crucial." > testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ ls
Android  dev       Music     PycharmProjects  StudioProjects  Videos
c++      Documents  Pictures  python           Templates
Desktop  Downloads  Public    snap             testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ cat testfile.txt
The enviroment is changing rapidly.
Protect the enviroment for future generations.
Enviroment awareness is crucial.
syedqasimali@syedqasimali-Inspiron-5559:~$ sed 's/enviroment/environment/g' testfile.txt
The environment is changing rapidly.
Protect the environment for future generations.
```

e. Number the lines in the file:

To number the lines in the output of the previous question using nl, you can pipe the corrected text into **nl** after using **sed**.

- First, correct the misspelling using the **sed** command.
- Use the **nl** command to add line numbers to the corrected output.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ sed 's/enviroment/environment/g' testfile.txt | nl
     1  The environment is changing rapidly.
     2  Protect the environment for future generations.
     3  Enviroment awareness is crucial.
```

Question: 02

a. Create an empty file and monitor it using the tail -f command. From another terminal, add lines to the file using a command like:
echo "testing" >> filename

b. After adding lines to the file, use the tr command to replace all occurrences of the letters A-F with the numbers 0-5.

c. View the binary content of the ls command (located at /bin/ls) using less. Use the -f option with less to force it to display the file, even though it's not text.

d. View the binary again using the od command. Try using it in default mode and with the options for outputting in hexadecimal.

Answer:

a. Creating an empty file and adding lines in it from another terminal:

- Create an empty file using **touch**.
- Use **tail-f** command to monitor the file, **tail-f** command leaves the terminal open and monitors any new lines added to the file and displays them.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ touch monitored_file.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ ls
Android    Downloads          Public          Templates
c++        monitored_file.txt PycharmProjects testfile1.txt
Desktop    monitor_file.txt   python          testfile.txt
dev        Music              snap            Videos
Documents  Pictures           StudioProjects
syedqasimali@syedqasimali-Inspiron-5559:~$ tail -f monitored_file.txt
This is a test line
Another line is added
```

b. 'tr' command:

- **Tr** command is used to transform letters into numbers.
- Use **tr** command to transform letters A-F into numbers 0-5.
- **cat monitored_file.txt**: Outputs the content of the file.
- **tr 'A-Fa-f' '0-50-5':** Translates:
  - Uppercase letters A-F to numbers 0-5.
  - Lowercase letters a-f to numbers 0-5.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ echo "This is a test line" >> monitored_file.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ echo "Another line is added" >> monitored_file.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ cat monitored_file.txt | tr 'A-Fa-f' '0-50-5'
This is 0 t4st lin4
@noth4r lin4 is 03343
```

c. Viewing Binary with 'od' command:

- Run **od** in its **default mode** (octal).
- Use the **-x** or **-t x1** option to view the content in **hexadecimal format**.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ od monitored_file.txt
0000000 064124 071551 064440 020163 020141 062564 072163 066040
0000020 067151 005145 067101 072157 062550 020162 064554 062556
0000040 064440 020163 062141 062544 005144
0000052
syedqasimali@syedqasimali-Inspiron-5559:~$ od -x monitored_file.txt
0000000 6854 7369 6920 2073 2061 6574 7473 6c20
0000020 6e69 0a65 6e41 746f 6568 2072 696c 656e
0000040 6920 2073 6461 6564 0a64
0000052
```

Question: 03

a. Use the split command to split the binary of the ls command into 1 KB chunks. Create a directory for the split files to make them easier to manage and delete.

b. Reassemble the split files and verify that the program still works. Ensure you are running the new copy (e.g., ./my_ls) and make it executable using the following command:

chmod a+rx my_ls

Answer:

a. Splitting the binary of ls commands using split commands:

- Find the full **path** of the **ls** command binary using **which.**
- Use the **split** command to split the binary file into **1 KB** chunks.
- **-b 1k**: Specifies a chunk size of 1 kilobyte (1 KB).
- **/bin/ls**: The binary file to be split.
- **split_ls_files/ls_chunk_**: The directory and prefix for the output files. Files will be named **ls_chunk_aa**, **ls_chunk_ab**, etc.
- List the contents of the **split_ls_files** directory to verify the chunks.

```
syedqasimali@syedqasimali-Inspiron-5559:~$ mkdir split_ls_files
syedqasimali@syedqasimali-Inspiron-5559:~$ which ls
/usr/bin/ls
syedqasimali@syedqasimali-Inspiron-5559:~$ split -b 1k /bin/ls split_ls_files/ls_chunk_
syedqasimali@syedqasimali-Inspiron-5559:~$ ls -lh split_ls_files/
total 540K
-rw-rw-r-- 1 syedqasimali syedqasimali 1.0K Nov 23 12:52 ls_chunk_aa
-rw-rw-r-- 1 syedqasimali syedqasimali 1.0K Nov 23 12:52 ls_chunk_ab
-rw-rw-r-- 1 syedqasimali syedqasimali 1.0K Nov 23 12:52 ls_chunk_ac
-rw-rw-r-- 1 syedqasimali syedqasimali 1.0K Nov 23 12:52 ls_chunk_ad
-rw-rw-r-- 1 syedqasimali syedqasimali 1.0K Nov 23 12:52 ls_chunk_ae
```

b. Reassembling the binary of ls:

- Use the **cat** command to concatenate the split files back together into a single file. Specify the directory and file prefix where the chunks were saved:
  cat split_ls_files/ls_chunk_* > my_ls
- Run the **chmod** command to mark the reassembled binary as executable.
  chmod a+rx my_ls
- **a+rx:** Adds read (r) and execute (x) permissions for all users.
- Run the reassembled binary explicitly using **./my_ls**
- Compare the outputs of the original and reassembled binaries. (**ls** & **./my_ls**)
- Verify the integrity of the reassembled binary by comparing checksums.
- If the checksums match, the reassembled binary is identical to the original. (**sha256sum /bin/ls my_ls**)

```
syedqasimali@syedqasimali-Inspiron-5559:~$ cat split_ls_files/ls_chunk_* > my_ls
syedqasimali@syedqasimali-Inspiron-5559:~$ chmod a+rx my_ls
syedqasimali@syedqasimali-Inspiron-5559:~$ ./my_ls
Android    Downloads           Pictures          split_ls_files  Videos
c++        monitored_file.txt  Public            StudioProjects
Desktop    monitor_file.txt    PycharmProjects   Templates
dev        Music               python            testfile1.txt
Documents  my_ls               snap              testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ ls
Android    Downloads           Pictures          split_ls_files  Videos
c++        monitored_file.txt  Public            StudioProjects
Desktop    monitor_file.txt    PycharmProjects   Templates
dev        Music               python            testfile1.txt
Documents  my_ls               snap              testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ ./my_ls
Android    Downloads           Pictures          split_ls_files  Videos
c++        monitored_file.txt  Public            StudioProjects
Desktop    monitor_file.txt    PycharmProjects   Templates
dev        Music               python            testfile1.txt
Documents  my_ls               snap              testfile.txt
syedqasimali@syedqasimali-Inspiron-5559:~$ sha256sum /bin/ls my_ls
12a6d908a68ccf6f9f3d799705577c28763f5deef6eddcff7643d6d8a6de543d  /bin/ls
12a6d908a68ccf6f9f3d799705577c28763f5deef6eddcff7643d6d8a6de543d  my_ls
```

Conclusion:

Both the original (ls) and reassembled binaries(./my_ls) the checksum binary is also identical hence, both binaries are same and isn't corrupted after splitting into chunks and reassembling.